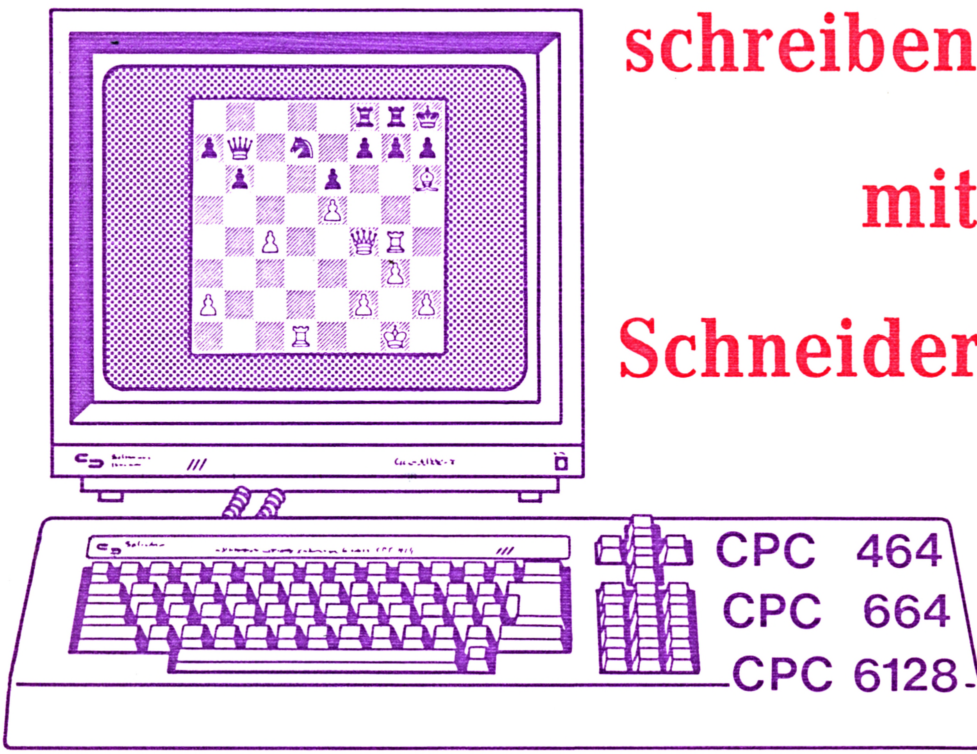
G.O. Hamann · J.-J. Eden

BASIC-Schachprogramm



schreiben mit Schneider

CPC 464
CPC 664
CPC 6128

DBV DEUTSCHER BETRIEBSWIRTE-VERLAG GMBH

G.O. Hamann · J.-J. Eden

BASIC-Schachprogramm schreiben mit Schneider

BASIC-Schachprogramm schreiben mit Schneider

Prof. Dr. G.O. Hamann · J.-J. Eden

DBV DEUTSCHER BETRIEBSWIRTE-VERLAG GMBH

Satz: GOHM ETXII & COMTRADE-Drucker
Druck- und Bindearbeit: Greiserdruck Rastatt
ISBN: 3-88640-035-2

VORWORT

Um insbesondere dem Schachspiel-Anfänger jederzeit einen Partner zur Verfügung zu stellen, werden sowohl zahlreiche Schachcomputer als auch Schachprogramme angeboten. Sie gewähren jedoch keinen Einblick in die Ablauflogik, die vor allem für Besitzer eines Personal Computers von Interesse ist. Diese Lücke soll das vorliegende Buch schließen. Es bietet

- das komplette Listing eines BASIC-Schachprogramms für die Personal Computer CPC 464, CPC 664 und CPC 6128 der Firma Schneider,

- ausführliche Erläuterungen zur Schachprogrammierung und zum BASIC-Schachprogramm,

- Hinweise für Änderungen, Ergänzungen und Verbesserungen.

Der Reiz dieses Buches liegt also weniger in der Möglichkeit, die Ausgaben für einen "spezialisierten" Schachcomputer oder ein teures Schachprogramm zu sparen. Der persönliche Gewinn ist vielmehr und vor allem darin zu sehen, daß das Schachspiel der Maschine seinen "black-box"-Charakter verliert, d. h. der Spieler durchschaut mit der Übernahme des Programms die Strategie des Computers. Und wenn diese dem Anwender nicht behagt, kann er sie sogar verändern, weil BASIC eine Programmiersprache ist, die von fast allen Hobbyisten der Datenverarbeitung beherrscht wird.

Im Hinblick auf die oben festgelegte Zielsetzung des Buches haben wir bewußt und konsequent darauf verzichtet, wichtige zeitkritische Routinen in Maschinensprache zu verfassen. Wer sie be-

herrscht, wird kaum größere Schwierigkeiten haben, nachträglich entsprechende Änderungen vorzunehmen. Das gleiche gilt für jene Leser, die ein Schachprogramm in einer anderen höheren Programmiersprache - wie z. B. PASCAL - schreiben möchten.

Unser Dank gebührt den Herren Rudolf Fessel, Volker Stürcken und Klaus Woelke für die freundliche Unterstützung.

Schortens und Waddewarden im April 1986

Günter O. Hamann
Jan-Jürgen Eden

Hinweis:

Um verhängnisvolle Setzfehler zu umgehen, wurden sämtliche in diesem Buch wiedergegebenen Programmzeilen/Programmausschnitte per LIST-Kommando auf einem an den Schneider-Computer angeschlossenen Typenrad-Drucker ausgedruckt und so direkt als Druckvorlage verwendet. Wenn daher das vom Leser übernommene Programm nach der Eingabe nicht ordnungsgemäß funktioniert, so sollte man den Fehler nicht in den gedruckten Listings des Buches suchen (vgl. hierzu auch Kapitel 5. Überpüfung des abgeschriebenen Programms !).

Um die Übersichtlichkeit innerhalb der Programmzeilen zu erhöhen, nutzten wir die folgende Eigenschaft der von den Rechnern der Firma Schneider verwendeten BASIC-Version: Wenn Programmzeilen im Kleinschriftmodus eingegeben werden, so wandelt das System die BASIC-Wörter in Großbuchstaben um, während die Variablennamen mit Kleinbuchstaben wiedergegeben werden. Um dann aber eine Verwechslung des Kleinbuchstaben "l" mit der Ziffer "1" zu vermeiden, haben wir bei den Variablennamen den Kleinbuchstaben "l" nur ein eiziges Mal benutzt, nämlich für die häufig verwendete Variable **fl** (Abkürzung für das englische Wort "**flag**" (= Merker, Schalter).

INHALTSVERZEICHNIS

Seite

Seite

1. Vor- und Nachteile der Schachprogrammierung in BASIC

Die Realisierung eines Schachprogramms in einer Interpreter-Sprache wie BASIC ist in erster Linie deshalb problematisch, weil die Ausführungszeiten verhältnismäßig lang werden.

Diesem Nachteil stehen aber gewichtige Vorteile gegenüber:

(1) Da die meisten privaten Besitzer eines Computers die Programmiersprache BASIC beherrschen, können sie die Logik eines in BASIC verfaßten Schachprogramms nachvollziehen.

(2) Das Nachvollziehen und Verstehen der vorgegebenen Logik eröffnet dann die Möglichkeit, das Programm

- zu verändern,
- zu verbessern,
- zu erweitern und
- zu korrigieren.

was im Interpreter-Betrieb der Programmiersprache BASIC besonders einfach ist.

Darüber hinaus läßt sich der anfänglich erwähnte Nachteil der verhältnismäßig langen Ausführungszeiten bei sehr vielen Rechnern dann aufheben, wenn - wie für die Systeme der Firma Schneider (= Amstrad) - Compiler angeboten werden, mit deren Hilfe man das

mit dem Interpreter erstellte und getestete BASIC-Programm in ein Maschinenprogramm übersetzen kann, das wesentlich schneller arbeitet.

2. Erläuterungen zum BASIC-Schachprogramm

2.1. Darstellung des Schachbretts

Das als bekannt vorausgesetzte Schachbrett mit den Figuren in der Ausgangsstellung hat folgendes Aussehen:

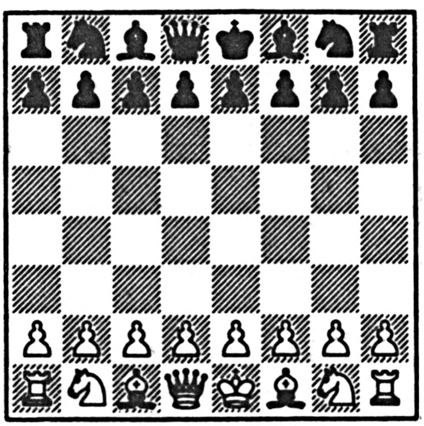


Abb. 1: Schachbrett mit Figuren in der Grundstellung

Das Schachbrett ist demnach so zu legen, daß das linke Eckfeld schwarz ist. Entsprechend einer internationalen Norm ist jedes Feld durch eine Buchstabe/Ziffer-Kombination bestimmt; beispielsweise hat das Feld unten links mit dem weißen Turm die Bezeichnung a1, das Feld mit dem weißen König die Bezeichnung e1, das

Feld mit dem schwarzen König die Bezeichnung e8 und das Feld oben rechts mit dem schwarzen Turm die Bezeichnung h8.

Zusammenfassend ergeben sich also die folgenden Benennungen der einzelnen Schachbrettfelder:

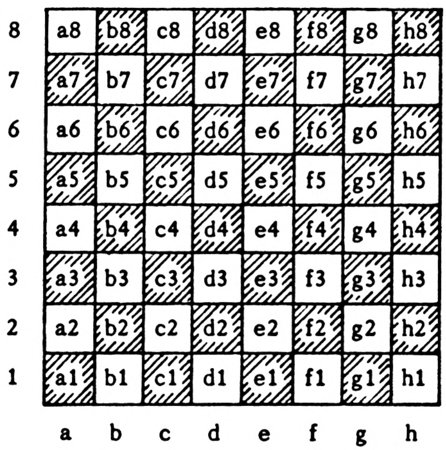


Abb. 2: Schachbrett mit den üblichen Feldbenennungen

Die erläuterte Darstellung eines Schachbretts ließe sich in gleicher Weise auch in einem Computer-Programm verwenden. Es hat sich allerdings als zweckmäßig erwiesen, die folgende Form zu verwenden, bei der die Buchstaben durch Ziffern ersetzt werden:

8	18	28	38	48	58	68	78	88
7	17	27	37	47	57	67	77	87
6	16	26	36	46	56	66	76	86
5	15	25	35	45	55	65	75	85
4	14	24	34	44	54	64	74	84
3	13	23	33	43	53	63	73	83
2	12	22	32	42	52	62	72	82
1	11	21	31	41	51	61	71	81
	1	2	3	4	5	6	7	8

Abb. 3: Schachbrett mit den computer-internen Feldbenennungen

In Anlehnung an unsere Beispiele auf den Seiten 2-01 und 2-02 hat dann das Feld unten links mit dem weißen Turm die Bezeichnung 11, das Feld mit dem weißen König die Bezeichnung 51, das Feld mit dem schwarzen König die Bezeichnung 58 und das Feld mit dem schwarzen Turm oben rechts die Bezeichnung 88. (Für die Zugberechnung werden wir das soeben erläuterte Feld allerdings noch erweitern; vgl. hierzu Seite 2-15, **Abb. 6:** Erweitertes computer-internes Schachbrett!)

Abweichend von einer eingängigeren Abkürzung der Figuren, bei der beispielsweise **d** für **D**ame, **k** für **K**önig, **t** für **T**urm verwendet wird, stellt man in einem Programm auch die Figuren mittels Ziffern dar, nämlich

1	für	Bauer (in Grundposition),
2	für	Bauer (bereits gezogen),
3	für	Springer,
4	für	Läufer,
5	für	Turm (noch rochadefähig),
6	für	Turm (bereits gezogen),
7	für	Dame
8	für	König (noch rochadefähig),
9	für	König (bereits gezogen)

In dem von uns realisierten Programm unterscheiden wir die Figuren der beiden Parteien (weiße Figuren einerseits und schwarze andererseits) lediglich durch ein Vorzeichen, und zwar sind jene Ziffern, welche **die weißen Figuren des Spielers** repräsentieren, **negativ; jene des Rechners (schwarz)** hingegen **ohne Vorzeichen (= positiv).**

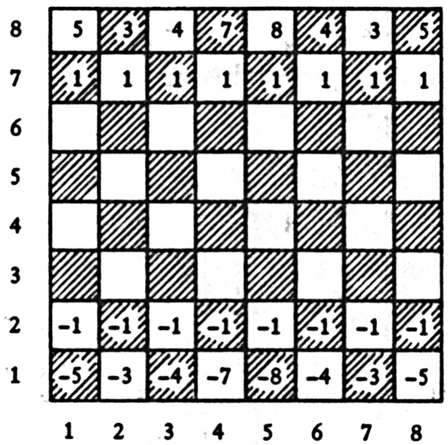


Abb. 4: Schachbrett mit der computer-internen Grundstellung

Abweichend von der erläuterten internen Darstellung erfolgt die Wiedergabe auf dem Bildschirm z u n ä c h s t mittels der bekannten Abkürzungen, wobei die Figuren des Rechners als Kleinbuchstaben (z. B. **d** für **D**ame, **k** für **K**önig, **t** für **T**urm) und jene des Spielers als Großbuchstaben (z. B. **D** für **D**ame, **K** für **K**önig, **T** für **T**urm) abgebildet werden. Im Kapitel 4.1 werden wir dann eine Möglichkeit aufzeigen, wie man die Buchstaben durch Schachfiguren ersetzen kann.

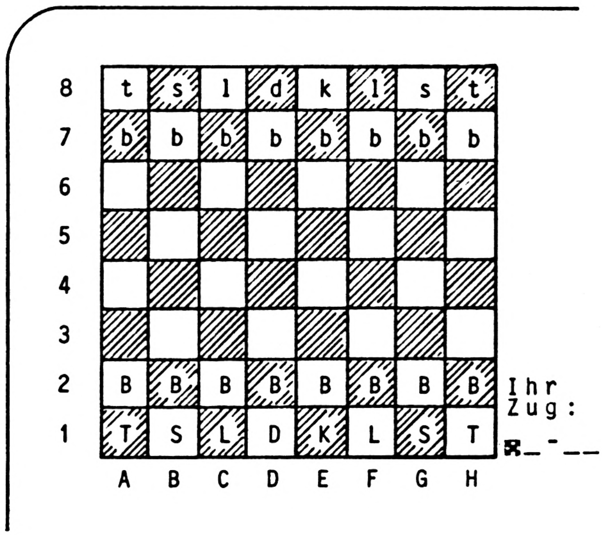


Abb. 5: Schachbrett auf dem Bildschirm mit "Figuren" in der Grundstellung

2.2. Spielregeln für das BASIC-Schachprogramm

Für das Spiel mit dem BASIC-Schachprogramm gelten die allgemeinen Schachregeln. Wenn man beispielsweise von C3 nach E5 ziehen will, dann muß **C3-E5 + ENTER** eingegeben werden. Falls man sich vertippt (und die **ENTER**-Taste noch nicht gedrückt) hat, ist der fehlerhafte Zug mit der **DEL**-Taste zu löschen. Anschließend kann man den gewünschten Zug neu eingeben.

Die kleine und die große Rochade spezifiziert man mit Hilfe des Königszugs (**+ ENTER**) - das Programm quittiert die Forderung des Spielers mit **0-0** (= kleine Rochade) bzw. mit **0-0-0** (= große Rochade).

Im übrigen verweisen wir auf die ergänzenden Bedienungshinweise in den Kapiteln 4.2., 4.4., 4.5. und 4.8. !

2.3. Abschnittsweise Erläuterungen zum Listing

Bei unseren Ausführungen zum Listing orientieren wir uns an der Reihenfolge der Befehle. Diese Vorgehensweise bietet sich an, weil

- das Programm ohnehin systematisch aufgebaut ist und weil

- so das Nachvollziehen der angestellten Überlegungen für den Leser erleichtert wird.

Programmausschnitt (1):

```
10 REM ********************************
20 REM *                              *
30 REM *   Schachprogramm in BASIC    *
40 REM *                              *
50 REM ********************************
60 REM
```

Erläuterungen zum Programmausschnitt (1):

Die **REM**-Befehle der Zeilen 10 - 60 bilden eine "Überschrift" und dienen lediglich der übersichtlichen Gestaltung des Programms.

Programmausschnitt (2):

```
70 REM ********************************
80 REM *        Steuerleiste          *
90 REM ********************************
100 :
200 GOSUB 1000   : REM Initialisierung
210 :
300 WHILE st=-1
310 GOSUB 2000   : REM Spielereingabe
320 GOSUB 6000   : REM Zugausgabe
330 WEND
340 :
400 WHILE st=1
410 GOSUB 4000   : REM Computerzug
420 GOSUB 6000   : REM Zugausgabe
430 WEND
440 :
500 IF st <> 0 GOTO 300
510 :
600 GOSUB 9000   : REM Spielende
610 :
700 END
```

Erläuterungen zum nebenstehenden Programmausschnitt (2):

Die Zeilen 70 - 700 (= **Steuerleiste**) bilden das Hauptprogramm, aus dem folgende Funktionsmodule aufgerufen werden:

- zu Beginn des Programmlaufs die "**Initialisierung**", die der Grundeinstellung von Variablen dient; anschließend

- die "**Spielereingabe**" und die "**Zugausgabe**", wenn der Merker **st** (= **Status**) den Wert -1 hat (Mittels der Variablen st besteht die Möglichkeit, dem Bediener des Systems die Entscheidung zu überlassen, ob der Computer oder der Anwender das Spiel eröffnet; vgl. hierzu Kap. 4.5!),

- "**Computerzug**" und "**Zugausgabe**", sofern **st** den Wert 1 (entspricht +1) hat, und

- bei einer Patt- oder Mattsituation das "**Spielende**" - der Merker **st** hat dann den Wert 0.

Durch den IF-Befehl in Zeile 500 bewirken wir, daß die Funktionsmodule "**Spielereingabe**, "**Zugausgabe**, "**Computerzug**" und "**Zugausgabe**" so lange wiederholt werden, bis das Spielende erreicht ist.

Programmausschnitt (3):

```
990  REM ******************************
1000 REM *       Initialisierung      *
1010 REM ******************************
1020 :
1030 DEFINT a, b, c, f, j, m, n, p, s
1040 DEFINT v, x, y
1050 DEFSTR i, g
1060 :
1070 DIM vm(10), f(99), jp(8), jm(9,2)
1080 DIM nj(9), pm(8), g(18), fm(120,3)
1090 DIM mm(27)
1100 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (3):

Im Funktionsmodul **"Initialisierung"**

- definieren wir alle im Programm verwendeten numerischen Variablen, und zwar mit den Anfangsbuchstaben a, b, c, f, j, m, n, p, s, v, x und y, als Ganzzahlvariablen (der Zugriff zu ihnen erfolgt wesentlich schneller als zu den "normalen" numerischen Variablen) und die Variablen mit den Anfangsbuchstaben i und g als Stringvariablen. - Wir reservieren mit

- **DIM vm(10), ...**
  Speicherplätze für Figurenwerte (**v**alue of **m**ate).
  (Bekanntlich haben die einzelnen Schachfiguren einen unterschiedlichen strategischen Wert für das Spiel. Der König ist in jedem Fall die wichtigste Figur, der Bauer hingegen hat die geringste Bedeutung.) Mit

- **DIM ..., f(99), ...**
  erzeugen wir die einzelnen Felder des Schachbretts (**f**ield). (Abweichend von unserer Darstellung auf Seite 2-03 reservieren wir nicht nur die Plätze für die eigentlichen Schachbrettfelder, sondern darüber hinaus 36 zusätzliche Felder, die das Schachbrett "einrahmen". Wir werden diese Felder benötigen, um sie mit sog. Scheinfiguren zu besetzen. Sie werden verhindern, daß der Computer bei der Berechnung seines jeweiligen Zuges mit der Figur das Schachbrett "verläßt".)

Programmausschnitt (3):

```
990  REM ******************************
1000 REM *       Initialisierung      *
1010 REM ******************************
1020 :
1030 DEFINT a, b, c, f, j, m, n, p, s
1040 DEFINT v, x, y
1050 DEFSTR i, g
1060 :
1070 DIM vm(10), f(99), jp(8), jm(9,2)
1080 DIM nj(9), pm(8), g(18), fm(120,3)
1090 DIM mm(27)
1100 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (3) / Forts.:

09	19	29	39	49	59	69	79	89	99
08	18	28	38	48	58	68	78	88	98
07	17	27	37	47	57	67	77	87	97
06	16	26	36	46	56	66	76	86	96
05	15	25	35	45	55	65	75	85	95
04	14	24	34	44	54	64	74	84	94
03	13	23	33	43	53	63	73	83	93
02	12	22	32	42	52	62	72	82	92
01	11	21	31	41	51	61	71	81	91
00	10	20	30	40	50	60	70	80	90

Abb. 6: Erweitertes computer-internes Schachbrett

Mit

- **DIM ..., jp(8), ...**
  schaffen wir Plätze, die Feldabstände aufnehmen werden (jump). Wir verwenden diese, um Zielkoordinaten einer Figur aus den Startkoordinaten zu berechnen. Die durch

- **DIM ..., jm(9,2)**
  reservierten Variablen (pointer for finding the correct numbers to determine the permitted jumps of a mate) sollen Werte aufnehmen, mit deren Hilfe es möglich sein wird, die für die jeweilige Figur festgelegten Feldabstände aus der Tabelle jp(..) für die Berechnung eines Zuges zu entnehmen.

Programmausschnitt (3):

```
990  REM *****************************
1000 REM *      Initialisierung      *
1010 REM *****************************
1020 :
1030 DEFINT a, b, c, f, j, m, n, p, s
1040 DEFINT v, x, y
1050 DEFSTR i, g
1060 :
1070 DIM vm(10), f(99), jp(8), jm(9,2)
1080 DIM nj(9), pm(8), g(18), fm(120,3)
1090 DIM mm(27)
1100 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (3) / Forts.:

Mit

- **DIM nj(9), ...**
  erzeugen wir Speicherplätze, denen wir an späterer Stelle Werte zuweisen, durch die die Anzahl der höchstzulässigen Schritte (number of jumps) einer bestimmten Figur festgelegt ist (beispielsweise darf der König nur einen, die Dame hingegen bis zu sieben Schritte bewegt werden).
  Die durch

- **DIM ..., pm(8), ...**
  erzeugten Variablen werden mögliche Positionen der jeweils untersuchten Figur (position of a mate) aufnehmen:

  in pm(1) die Startkoordinate der Figur,
  in pm(2) die Zielkoordinate der Figur,
  in pm(3) die Koordinate eines "en passant" geschlagenen Bauern - dann ist
  in pm(4) der Wert 0,

  o d e r

  in pm(3) die Startkoordinate des Turmzugs und
  in pm(4) dessen Zielkoordinate, allerdings nur bei Ausführung einer Rochade
  (Wenn weder ein "en-passant-Schlag" noch eine Rochade möglich ist, dann haben pm(3) und pm(4) den Wert 0.),
  in pm(5) und
  in pm(6) und
  in pm(7) und
  in pm(8) die zu sichernden Werte der Variablen pm(1) bis pm(4), und zwar nur bei Spielstärke 2, wenn der Computer den fiktiven Gegenzug des Spielers erzeugt, um seinen eigenen im Speicher zu erhalten.
  Mit der Anweisung

Programmausschnitt (3):

```
990  REM ******************************
1000 REM *      Initialisierung       *
1010 REM ******************************
1020 :
1030 DEFINT a, b, c, f, j, m, n, p, s
1040 DEFINT v, x, y
1050 DEFSTR i, g
1060 :
1070 DIM vm(10), f(99), jp(8), jm(9,2)
1080 DIM nj(9), pm(8), g(18), fm(120,3)
1090 DIM mm(27)
1100 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (3) / Forts.:

- **DIM ..., g(18), ...**
  erzeugen wir Speicherplätze (**g**raphic characters), in denen die Druckzeichen der Figuren für den Bildschirm abzulegen sein werden.
  Die durch

- **DIM ..., fm(120,3)**
  reservierten Variablen (**f**ield of **m**ates) sollen dazu dienen, Startkoordinaten in fm(..,1), Zielkoordinaten in fm(..,2) und die Bewertung der jeweiligen Stellung in fm(..,3) aufzunehmen.
  Wir reservieren mit

- **DIM mm(27)**
  Speicherplätze (possible **m**oves of a **m**ate), in denen bei Spielstärke 2 die möglichen Zielkoordinaten e i n e r Figur des Spielers gespeichert werden. (Die Speicherung der Startkoordinate ist hier nicht erforderlich, weil eine ausgesuchte Figur nur eine Startkoordinate haben kann.)

Programmausschnitt (4):

```
1110 FOR a=0 TO 9 : READ vm(a) : NEXT a
1120 DATA 1, 5, 5, 15, 15, 25, 25
1130 DATA 50, 2000,2000
1140 :
1150 FOR a=0 TO 99 : READ f(a) : NEXT a
1160 DATA 10,10,10,10,10,10,10,10,10,10
1170 DATA 10,-5,-1, 0, 0, 0, 0, 1, 5,10
1180 DATA 10,-3,-1, 0, 0, 0, 0, 1, 3,10
1190 DATA 10,-4,-1, 0, 0, 0, 0, 1, 4,10
1200 DATA 10,-7,-1, 0, 0, 0, 0, 1, 7,10
1210 DATA 10,-8,-1, 0, 0, 0, 0, 1, 8,10
1220 DATA 10,-4,-1, 0, 0, 0, 0, 1, 4,10
1230 DATA 10,-3,-1, 0, 0, 0, 0, 1, 3,10
1240 DATA 10,-5,-1, 0, 0, 0, 0, 1, 5,10
1250 DATA 10,10,10,10,10,10,10,10,10,10
1260 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (4):

Den in Zeile 1070 reservierten Plätzen **vm(..)** für Figurenwerte werden nun durch die Befehle der Zeilen 1110 bis 1130 die folgenden "Gewichte" zugeordnet:

Variable	Wert	Feldbelegung
vm(0)	1	0 (= leeres Feld)
vm(1)	5	1 (= Bauer in Grundposition)
vm(2)	5	2 (= Bauer bereits gezogen)
vm(3)	15	3 (= Springer)
vm(4)	15	4 (= Läufer)
vm(5)	25	5 (= Turm, noch rochadefähig)
vm(6)	25	6 (= Turm, bereits gezogen)
vm(7)	50	7 (= Dame)
vm(8)	2000	8 (= König, noch rochadefähig)
vm(9)	2000	9 (= König, bereits gezogen)

In den Zeilen 1150 bis 1250 weisen wir den Feldern des Schachbrettrahmens sog. Scheinfiguren (= 10) zu (vgl. hierzu unsere Ausführungen auf Seite 2-13!) zu.

Außerdem "stellen wir die Figuren", die durch die auf Seite 2-04 erwähnten Ziffern repräsentiert werden, in ihre Grundposition, wie wir sie bereits in **Abb. 4** auf Seite 2-05 dargestellt haben.

Programmausschnitt (5):

```
1270 FOR a=1 TO 8 : READ jp(a) : NEXT a
1280 DATA 9, 11, 10, 1, 12, 21, 8, 19
1290 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (5):

Den indizierten Variablen jp(1) ... jp(8) weisen wir an dieser Stelle Feldabstandswerte zu, die wir an einer späteren Stelle des Programms zur Ermittlung eines figurgemäßen Zuges benötigen. Die ersten beiden Zahlen dienen zur Berechnung der Läuferzüge, die beiden folgenden für jene des Turms und die vier letzten für jene des Springers. Darüber hinaus werden wir die vier ersten Zahlen (= Läufer + Turm) zur Ermittlung der Königs- und der Damenzüge verwenden. Für den Bauern erübrigt sich hier eine Festlegung, da die Variationsbreite der Zugmöglichkeiten zu gering ist.

Beispiel:

Wir unterstellen, der Springer des Rechners befände sich auf dem Feld 65 (vgl. **Abb. 6** auf Seite 2-15!), was der Position f5 entspricht. Mit Hilfe der letzten vier Zahlen (12, 21, 8, 19) lassen sich nunmehr mögliche Springerzüge berechnen.

Anders ausgedrückt:
Ausgehend von der Startkoordinate 65 lassen sich denkbare Zielkoordinaten folgendermaßen ermitteln:

Startkoordinate		Feldabstand		Zielkoordinate
65	+	12	→	77
65	+	21	→	86
65	+	8	→	73
65	+	19	→	84
65	-	12	→	53
65	-	21	→	44
65	-	8	→	57
65	-	19	→	46

Programmausschnitt (6):

```
1300 FOR a=1 TO 9 : READ nj(a) : NEXT a
1310 DATA 1, 1, 1, 7, 7, 7, 7, 1, 1
1320 :
1330 FOR a=1 TO 9 : FOR b=1 TO 2
1340 READ jm(a,b)
1350 NEXT b, a
1360 DATA 1,4, 1,4, 5,8, 1,2, 3,4
1370 DATA 3,4, 1,4, 1,4, 1,4
1380 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (6):

Wie wir bereits auf Seite 2-17 ausführten, dienen die Speicherplätze nj(..) dazu, die Anzahl der höchstzulässigen Schritte einer bestimmten Figur aufzunehmen. Die entsprechenden Zuweisungen, nämlich

1	für	Bauer (in Grundposition)
1	für	Bauer (bereits gezogen)
1	für	Springer
7	für	Läufer
7	für	Turm (noch rochadefähig)
7	für	Turm (bereits gezogen)
7	für	Dame
1	für	König (noch rochadefähig)
1	für	König (bereits gezogen)

erfolgen in den Zeilen 1300 bis 1310.

Programmausschnitt (6):

```
1300 FOR a=1 TO 9 : READ nj(a) : NEXT a
1310 DATA 1, 1, 1, 7, 7, 7, 7, 1, 1
1320 :
1330 FOR a=1 TO 9 : FOR b=1 TO 2
1340 READ jm(a,b)
1350 NEXT b, a
1360 DATA 1,4, 1,4, 5,8, 1,2, 3,4
1370 DATA 3,4, 1,4, 1,4, 1,4
1380 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (6) / Forts.:

Die Speicherplätze jm(..,..) werden in den Zeilen 1330 bis 1370 mit jenen Werten versorgt, mit deren Hilfe es möglich sein wird, die für die jeweilige Figur festgelegten Feldabstände aus der Tabelle jp(..) für die Berechnung eines Zuges zu entnehmen (vgl. hierzu auch Seite 2-15!). Wir wollen die Funktionsweise anhand des folgenden Beispiels verdeutlichen.

Beispiel:

Es sollen die Feldabstandswerte für den Springer ermittelt werden. Mit Hilfe der für diese Figur (Springer = **3**, vgl. Seite 2-04!) spezifischen Variablen jm(**3**,1) und jm(**3**,2) - sie enthalten die Zahlen 5 und 8 - begrenzen wir (an späterer Stelle des Programms) in der Tabelle jp(..) den ersten (hier die 5. Zahl) und den letzten (hier die 8. Zahl) Feldabstandswert. Demnach lauten die für den Springer zu verwendenden Feldabstandswerte 12, 21, 8 und 19 (vgl. Zeile 1280 im Programmausschnitt (5) auf Seite 2-22!).

Programmausschnitt (7):

```
1390 FOR a=0 TO 18 : READ g(a) : NEXT a
1400 DATA K, K, D, T, T, L, S, B, B, ""
1410 DATA b, b, s, l, t, t, d, k, k
1420 :
1430 MODE 1
1440 INK 1, 24
1450 INK 2, 8
1460 :
1470 PRINT TAB(9) "Schneider-";
1480 PRINT "Schachprogramm"
1490 LOCATE 16, 3
1500 PRINT "in BASIC"
1510 LOCATE 12, 5
1520 PRINT "******************"
1530 LOCATE 3, 11
1540 PRINT "Bitte waehlen Sie ..."
1550 LOCATE 3, 15
1560 PRINT "Spielstaerke 1"
1570 LOCATE 3, 16
1580 PRINT "(ca.  3 Minuten ";
1590 PRINT "bis zum Zug)  ... (1)"
1600 LOCATE 3, 18
1610 PRINT "Spielstaerke 2"
1620 LOCATE 3, 19
1630 PRINT "(ca. 10 Minuten ";
1640 PRINT "bis zum Zug)  ... (2)"
1650 LOCATE 14, 23
1660 PRINT "... ´1´ oder ´2´ druecken!"
1670 in = INKEY$
1680 IF in < "1" OR in > "2" GOTO 1670
1690 mo = VAL(in)
1700 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (7):

In den Zeilen 1390 bis 1410 weisen wir den Variablen g(0) ... g(18) die folgenden Zeichen zu:

K, K, D, T, T, L, S, B, B, "" ← (Leerstring)
b, b, s, l, t, t, d, k, k.

Eine Beziehung zwischen den intern als Ziffern dargestellten Figuren, nämlich

-9, -8, -7, -6, -5, -4, -3, -2, -1, 0 (= leeres Feld),
1, 2, 3, 4, 5, 6, 7, 8, 9

einerseits und den obigen Druckzeichen andererseits läßt sich dadurch herstellen, daß man zu der als Ziffer repräsentierten Figur den Wert 9 addiert, um so den gewünschten Index für g(..) zu erhalten. Diese etwas umständliche Vorgehensweise ist deshalb erforderlich, weil negative Indizes nicht zulässig sind.

Mit Hilfe der Anweisung in Zeile 1430 ändern wir den Bildschirmmodus auf 1 (= 40 Zeichen pro Zeile, maximal 4 Farben), während die beiden folgenden Befehle die Hintergrundfarbe (für das Schachbrett) und die Zeichenfarbe (für die "Figuren" und die auszugebenden Nachrichten) festlegen.

Die Anweisungen der Zeilen 1470 bis 1690 erzeugen das Menue-Bild zu Beginn des Programms, das dem Anwender die Auswahl zwischen zwei Spielstärken ("1" oder "2") ermöglicht. Die eingegebene Spielstärke wird als numerischer Wert im Merker **mo** (**mo**de) abgelegt, um damit später den Programmablauf zu steuern.

Programmausschnitt (8):

```
1710 CLS
1720 FOR a=1 TO 8
1730 PRINT
1740 PRINT CHR$( 57 - a )
1750 PRINT
1760 NEXT a
1770 TAG
1780 MOVE 9, 12
1790 FOR a=1 TO 8
1800 PRINT "  ";
1810 PRINT CHR$( 64 + a ); " ";
1820 NEXT a
1830 TAGOFF
1840 FOR a=10 TO 80 STEP 10
1850 FOR b=1 TO 8
1860 pf = a + b
1870 GOSUB 6500
1880 NEXT b, a
1890 :
1900 WINDOW#1, 35, 40, 21, 25
1910 WINDOW#2, 35, 40, 1, 19
1920 LOCATE#2, 1,19
1930 bm = 11
1940 st = -1
1950 ro = 0
1960 ma = 0
1970 RETURN
1980 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (8):

Durch die Befehle der Zeilen 1710 bis 1880 erzeugen wir das leere Schachbrett einschließlich der Randbeschriftung auf dem Bildschirm:

Zeilen 1720 - 1760: Ausgabe der Ziffern 1 bis 8 der Randbeschriftung,

Zeilen 1770 - 1830: Ausgabe der Buchstaben A bis H der Randbeschriftung,

Zeilen 1840 - 1880: Ausgabe der einzelnen Brettfelder mit den entsprechenden Figuren (durch Aufruf eines Unterprogramms in Zeile 1870, das in Zeile 6500 beginnt).

Darüber hinaus definieren wir in den Zeilen 1900 und 1910 zwei Fenster (windows):

WINDOW#1: Es bildet einen definierten und separat ansprechbaren Bildschirmbereich, in dem alle für das Spiel erforderlichen Nachrichten erscheinen, wie z. B.

```
Ihr          Ich
Zug:         werte:

B_-__        __-__
```

WINDOW#2: Es bildet einen weiteren separat ansprechbaren Bildschirmbereich, in dem die letzten 12 ausgeführten Züge wiedergegeben werden. (Zu diesem Zweck setzen wir den Cursor des WINDOW#2 in Zeile 1920 auf die unterste Zeile dieses Fensters.)

Wir setzen die an späterer Stelle (vgl. Seite 2-59!) zu erläuternde Variable **bm** (**b**est-**m**ove) auf den Anfangswert 11. Ferner erhält der Merker **st** Wert -1. Wir erreichen damit, daß der Bediener das Schachspiel eröffnen muß. Die Merker **ro** und **ma** erhalten den Wert 0 nur zur Verdeutlichung ihrer Funktion, und zwar dient ro zur Unterstützung der **Ro**chadenausführung, und ma wird zur Steuerung des Programmlaufs im **Ma**tt- oder Pattfall verwendet.

Programmausschnitt (9):

```
1990 REM ******************************
2000 REM *      Spielereingabe        *
2010 REM ******************************
2020 :
2030 WINDOW SWAP 1,0
2040 fl = 0
2050 ro = 0
2060 LOCATE 1,1
2070 PRINT "Ihr"
2080 PRINT " Zug: "
2090 PRINT CHR$(143); "_-__ "
2100 LOCATE 1,4
2110 a = 1
2120 in = INKEY$
2130 IF in = CHR$(127) GOTO 2260
2140 IF in < "a" OR in > "h" GOTO 2120
2150 PRINT UPPER$(in);
2160 PRINT CHR$(143); CHR$(8);
2170 pm(a) = ( ASC(in)-96 ) * 10
2180 in = INKEY$
2190 IF in = CHR$(127) GOTO 2260
2200 IF in < "1" OR in > "8" GOTO 2180
2210 PRINT UPPER$(in);
2220 IF a = 1 THEN PRINT "-";
2230 PRINT CHR$(143); CHR$(8);
2240 pm(a) = pm(a) + VAL(in)
2250 IF a = 1 THEN a = 2 : GOTO 2120
2260 IF in = CHR$(127) GOTO 2060
2270 pm(3) = 0
2280 pm(4) = 0
2290 in = INKEY$
2300 IF in = CHR$(127) GOTO 2040
2310 IF in <> CHR$(13) GOTO 2290
2320 PRINT " "
```

Erläuterungen zum nebenstehenden Programmausschnitt (9):

Mit der Anweisung in Zeile 2030 aktivieren wir den Bildschirmbereich WINDOW#1, so daß hierin - wenn keine Spezifikation für einen anderen Bereich vorliegt - zunächst alle weiteren Ausgaben erfolgen.

Anschließend erhalten die Merker **fl** (**flag**) und **ro** den Wert 0. (fl nimmt in diesem Funktionsmodul nur dann den Wert 1 an, wenn der Spieler eine unzulässige Zugeingabe getätigt hat. ro wird auf 1 gesetzt, wenn der Bediener die Ausführung eines Rochadezugs wünscht.)

In den Zeilen 2060 bis 2090 wird folgender Text ausgegeben:

```
Ihr
Zug:

▩_-__
```

Damit das vom Anwender eingegebene Zeichen auf der Stelle des Scheincursors (▩) erscheint, setzen wir mit der Anweisung in Zeile 2100 die Druckposition auf den Scheincursor zurück. Die Eingabe des ersten Zeichens erfolgt in Zeile 2120. Daran anschließend untersucht das System, ob **DEL** gedrückt wurde (Zeile 2130), um ggf. mit diesem Programmabschnitt erneut zu beginnen. Zu diesem Zweck (wenn also **DEL** tatsächlich betätigt wurde) erfolgt ein Sprung nach Zeile 2260, um von hier aus nach Zeile 2060 zurückzukehren. Andernfalls erfolgt die Überprüfung (Zeile 2140), ob das eingegebene Zeichen eine Spalte des Schachbretts (A - H) spezifiziert. Eine unzulässige Angabe hat einen Rücksprung nach Zeile 2120 zur Konsequenz. Mit dem folgenden Befehl (Zeile 2150) erreichen wir, daß der eingegebene Buchstabe ausgegeben wird.

Programmausschnitt (9):

```
1990 REM *****************************
2000 REM *      Spielereingabe       *
2010 REM *****************************
2020 :
2030 WINDOW SWAP 1,0
2040 fl = 0
2050 ro = 0
2060 LOCATE 1,1
2070 PRINT "Ihr"
2080 PRINT " Zug: "
2090 PRINT CHR$(143); "_-__ "
2100 LOCATE 1,4
2110 a = 1
2120 in = INKEY$
2130 IF in = CHR$(127) GOTO 2260
2140 IF in < "a" OR in > "h" GOTO 2120
2150 PRINT UPPER$(in);
2160 PRINT CHR$(143); CHR$(8);
2170 pm(a) = ( ASC(in)-96 ) * 10
2180 in = INKEY$
2190 IF in = CHR$(127) GOTO 2260
2200 IF in < "1" OR in > "8" GOTO 2180
2210 PRINT UPPER$(in);
2220 IF a = 1 THEN PRINT "-";
2230 PRINT CHR$(143); CHR$(8);
2240 pm(a) = pm(a) + VAL(in)
2250 IF a = 1 THEN a = 2 : GOTO 2120
2260 IF in = CHR$(127) GOTO 2060
2270 pm(3) = 0
2280 pm(4) = 0
2290 in = INKEY$
2300 IF in = CHR$(127) GOTO 2040
2310 IF in <> CHR$(13) GOTO 2290
2320 PRINT " "
```

Erläuterungen zum nebenstehenden Programmausschnitt (9) / Forts.:

Die beschriebene Eingabeform ermöglicht die kontrollierte Abfrage von Zeichen sowie deren Ausgabe und wird deshalb in analoger Weise auch für die Zeilenwerte verwendet (Zeile 2180 - 2210).

Die beschriebene Eingaberoutine (Schleife von Zeile 2110 - 2250) wird zweimal durchlaufen, nämlich sowohl für die Eingabe der Start- als auch für die Eingabe der Zielkoordinate eines Zuges. Mit den Befehlen der Zeilen 2170 und 2240 erfolgt die Umsetzung aus der üblichen Schachnotation in die rechnerinterne Darstellungsform.

Im Programm wird zunächst einmal unterstellt, daß kein "en-passant-Schlag" und keine Rochade eingegeben wurde. Deshalb sezten wir die entsprechenden Merker pm(3) und pm(4) auf 0.

Abweichend von der Prozedur bei der Eingabe der Start- und Zielkoordinaten wird hiernach überprüft, ob **DEL** oder **ENTER** betätigt wurde, um ggf. zurückzuverzweigen (erneute Eingabemöglichkeit) oder den Programmlauf fortzusetzen.

Programmausschnitt (10):

```
2330 IF f( pm(1) ) >= 0 GOTO 2040

2340 IF pm(1) <> 51 GOTO 2380

2350 IF f(51) <> -8 GOTO 2380

2360 IF ABS(pm(1)-pm(2)) <>20 GOTO 2380

2370 ro = 1

2380 m = -f( pm(1) )

2390 :

2400 ON ro + 1 GOSUB 3000, 7100

2410 IF fl = 1 GOTO 2040

2420 IF ro <> 0 THEN 2470

2430 GOSUB 6700

2440 GOSUB 7400

2450 GOSUB 6900

2460 IF fl = 1 GOTO 2040

2470 WINDOW SWAP 1,0

2480 RETURN

2490 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (10):

Im nebenstehenden Abschnitt beginnen wir mit der Zugüberprüfung. Zunächst untersuchen wir, ob es sich bei der zu ziehenden Figur um eine solche des Spielers handelt (Zeile 2330): f(pm(1)) muß negativ (<0) sein. Ist dies nicht der Fall, so hat der Spieler die Zugeingabe zu wiederholen (Rücksprung nach Zeile 2040).

Ob der Königszug der Rochade eingegeben wurde, untersuchen wir mit den Befehlen der Zeilen 2340 bis 2360. Ist dies der Fall, so weisen wir dem Merker ro den Wert 1 zu (Zeile 2370).

Um im weiteren Programmablauf nicht mehr mit dem komplexen Ausdruck **-f(pm(1))** arbeiten zu müssen, der die rechner-interne "Figur" des Spielers (ohne das negative Vorzeichen) repräsentiert, nehmen wir in Zeile 2380 eine Zuweisung nach **m** (**m**ate) vor.

Abhängig von ro + 1 verzweigen wir in Zeile 2400 in die Subroutinen Zugüberprüfung (ab Zeile 3000) bzw. Rochadenüberprüfung (ab Zeile 7100). Wenn das System dort feststellen muß, daß der eingegebene Zug nicht ausgeführt werden darf, erhält der Merker fl den Wert 1, mit der Folge, daß nach der Rückkehr (aus einer der beiden Subroutinen) in Zeile 2410 zum Anfang des Funktionsmoduls zur Neueingabe des Zuges zurückgekehrt wird. - Wurde in der Subroutine Rochadenüberprüfung festgestellt, daß die vom Spieler gewünschte Rochade zulässig ist, dann erhielt der Merker ro dort den Wert -10 (= große Rochade) oder +10 (= kleine Rochade). Dies hat zur Folge, daß nach der Rückkehr ins Funktionsmodul Spielereingabe in Zeile 2420 zum Modulende verzweigt wird. Andernfalls ist nur noch zu überprüfen, ob durch die Ausführung des Zuges - sie erfolgt für interne Zwecke in der Subroutine ab Zeile 6700 - der eigene König ins Schach geraten würde (Subroutine ab Zeile 7400). Nachdem der untersuchte Zug mit der Subroutine ab Zeile 6900 rückgängig gemacht worden ist, überprüfen wir durch Abfrage des Merkers fl in Zeile 2460, ob in der Subroutine ab Zeile 7400 eine Bedrohung des Königs festgestellt worden ist, um im Be-

Programmausschnitt (10):

```
2330 IF f( pm(1) ) >= 0 GOTO 2040

2340 IF pm(1) <> 51 GOTO 2380

2350 IF f(51) <> -8 GOTO 2380

2360 IF ABS(pm(1)-pm(2)) <>20 GOTO 2380

2370 ro = 1

2380 m = -f( pm(1) )

2390 :

2400 ON ro + 1 GOSUB 3000, 7100

2410 IF fl = 1 GOTO 2040

2420 IF ro <> 0 THEN 2470

2430 GOSUB 6700

2440 GOSUB 7400

2450 GOSUB 6900

2460 IF fl = 1 GOTO 2040

2470 WINDOW SWAP 1,0

2480 RETURN

2490 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (10) / Forts.:

jahungsfalle zum Anfang des Funktionsmoduls zur Neueingabe eines Zuges zurückzukehren.

Falls bei den Überprüfungen kein Regelverstoß festgestellt wurde (fl = 0), so wird der Zug als e i n zulässiger akzeptiert.

Mit der Anweisung in Zeile 2470 reaktivieren wir den Gesamtbildschirm als Ausgabeeinheit, um anschließend (Zeile 2480) in die Steuerleiste zurückzukehren.

Programmausschnitt (11):

```
3000 REM *** Zugueberpruefung ***
3010 :
3020 IF pm(2) = pm(1) GOTO 3190
3030 jp = 0
3040 FOR a=jm(m,1) TO jm(m,2)
3050 IF jp <> 0 GOTO 3090
3060 jp = jp(a)
3070 c = ( pm(2)-pm(1) ) MOD jp
3080 IF c <> 0 THEN jp = 0
3090 NEXT a
3100 IF jp = 0 GOTO 3190
3110 jp = jp * SGN( pm(2)-pm(1) )
3120 FOR a=pm(1)+jp TO pm(2)-jp STEP jp
3130 IF f(a) <> 0 THEN fl = 1
3140 NEXT a
3150 IF fl = 1 GOTO 3200
3160 IF f( pm(2) ) < 0 GOTO 3190
3170 IF m < 4 OR m > 7 THEN GOSUB 3300
3180 GOTO 3200
3190 fl = 1
3200 RETURN
3210 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (11):

Bisher wurde nur überprüft, ob ob bei der Zuganforderung das Startkoordinatenfeld tatsächlich eine Figur des Spielers enthält und ob eine Rochade ausgeführt werden soll.

Im nebenstehenden Programmausschnitt setzen wir die Überprüfungen für alle Züge - ausgenommen die Rochade - fort, um bei Feststellung eines Regelverstoßes zum Befehl der Zeile 3190 zu springen. Hier setzen wir ggf. den "Fehler-Merker" fl auf 1, um anschließend ins Funktionsmodul Spielereingabe zurückzukehren, wo eine erneute Zugeingabe eingeleitet wird.

Die Überprüfungen im einzelnen:

Zeile 3020:	Untersuchung, ob Start- und Zielkoordinate identisch sind,
Zeilen 3030 - 3090:	Berechnung der Schrittweite der zu bewegenden Figur, um in
Zeile 3100	die Zulässigkeit der Bewegungsart sicherzustellen,
Zeile 3110:	Berechnung der Bewegungsrichtung aus der Start- und Zielkoordinate, um in den
Zeilen 3120 - 3150	zu kontrollieren, ob bei diesem Zug keine Figur übersprungen wird,
Zeile 3160:	Untersuchung, ob auf dem Zielkoordinatenfeld eine eigene Figur steht,
Zeile 3170:	Einleitung von zusätzlichen Untersuchungen, wenn ein Bauer, Springer oder König gezogen werden soll.

Programmausschnitt (12):

```
3300 REM *** Sonderueberpruefungen ***
3310 :
3320 ON m GOSUB 3400, 3500
3330 IF m > 2 THEN GOSUB 3700
3340 RETURN
3350 :
3400 REM *** Bauer (1) ***
3410 :
3420 IF pm(2) - pm(1) <> 2 GOTO 3450
3430 IF f( pm(2) ) <> 0 THEN f1 = 1
3440 GOTO 3460
3450 GOSUB 3500
3460 RETURN
3470 :
3500 REM *** Bauer (2) ***
3510 :
3520 IF ABS( pm(2)-pm(1) )=10 GOTO 3600
3530 IF pm(2) - pm(1) <> 1 GOTO 3560
3540 IF f( pm(2) ) = 0 GOTO 3610
3550 GOTO 3600
3560 IF f( pm(2) ) > 0 GOTO 3610
3570 IF pm(1) MOD 10 <> 5 GOTO 3600
3580 pm(3) = pm(2) - 1
3590 IF f( pm(3) ) = 1 GOTO 3610
3600 f1 = 1
3610 GOSUB 3700
3620 RETURN
3630 :
3700 REM *** nur ein Schritt ! ***
3710 :
3720 IF pm(2) - pm(1) <> jp THEN f1 = 1
3730 RETURN
3740 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (12):

Wenn die Variable m, in der die zu ziehende Figur in Form einer Ziffer (vgl. Seite 2-04!) gespeichert ist, eine 1 (= Bauer in Grundposition) oder eine 2 (= Bauer, bereits gezogen) enthält, wird aufgrund der Anweisung in Zeile 3320 entweder die in Zeile 3400 oder die in Zeile 3500 beginnende Subroutine aufgerufen, um dort zu überprüfen, ob der gewünschte Bauernzug korrekt ist: Im Fehlerfalle erhält der Merker fl den Wert 1, um damit nach der Rückkehr ins Funktionsmodul Spielereingabe sicherzustellen, daß die Eingabe wiederholt wird.

Enthält die Variable m hingegen eine 3 oder 8 oder 9, rufen wir aufgrund des Befehls in Zeile 3330 die in Zeile 3700 beginnende Subroutine auf, mit deren Hilfe wir gewährleisten, daß der Springer oder König nur e i n e n der für die Figur zulässigen Schritte ausführen soll.

Die Überprüfungen im einzelnen:

Zeile 3420:	Untersuchung, ob ein Doppelschritt vorliegt; ist dies der Fall, kontrollieren wir in
Zeile 3430,	ob das Feld der Zielkoordinate unbesetzt ist.
Zeile 3450:	Aufruf der Routine ab Zeile 3500, denn für den Bauern in Grundposition gelten auch die für den bereits gezogenen Bauern bestehenden Regeln.
Zeile 3520:	Da für den Bauern (u. a.) jene Feldabstände mitverwendet werden, die auch für den Turm gelten, sorgen wir hier dafür, daß ein evtl. festgestellter Feldabstandswert 10 (= waagerechte Bewegung) zur Verwerfung des Zuges führt.

Programmausschnitt (12):

```
3300 REM *** Sonderueberpruefungen ***
3310 :
3320 ON m GOSUB 3400, 3500
3330 IF m > 2 THEN GOSUB 3700
3340 RETURN
3350 :
3400 REM *** Bauer (1) ***
3410 :
3420 IF pm(2) - pm(1) <> 2 GOTO 3450
3430 IF f( pm(2) ) <> 0 THEN fl = 1
3440 GOTO 3460
3450 GOSUB 3500
3460 RETURN
3470 :
3500 REM *** Bauer (2) ***
3510 :
3520 IF ABS( pm(2)-pm(1) )=10 GOTO 3600
3530 IF pm(2) - pm(1) <> 1 GOTO 3560
3540 IF f( pm(2) ) = 0 GOTO 3610
3550 GOTO 3600
3560 IF f( pm(2) ) > 0 GOTO 3610
3570 IF pm(1) MOD 10 <> 5 GOTO 3600
3580 pm(3) = pm(2) - 1
3590 IF f( pm(3) ) = 1 GOTO 3610
3600 fl = 1
3610 GOSUB 3700
3620 RETURN
3630 :
3700 REM *** nur ein Schritt ! ***
3710 :
3720 IF pm(2) - pm(1) <> jp THEN fl = 1
3730 RETURN
3740 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (12) / Forts.:

Zeile 3530:	Untersuchung, ob ein gerader Einzelschritt vorliegt; ist dies nicht der Fall, setzt der Programmlauf in Zeile 3560 fort, sonst kontrollieren wir in den
Zeilen 3540 - 3550,	ob das Feld der Zielkoordinate unbesetzt ist.
Zeile 3560:	Untersuchung, ob das Zielkoordinatenfeld eine gegnerische Figur enthält,
Zeilen 3570 - 3590:	Untersuchung, ob ein legaler "en-passant-Schlag" eingegeben wurde (ggf. weisen wir in Zeile 3580 der Variablen pm(3) die Koordinate des geschlagenen Bauern zu),
Zeile 3610:	Auruf einer in Zeile 3700 beginnenden Routine; dort untersuchen wir in
Zeile 3720,	ob der Bauer nur e i n e n der für ihn zulässigen Schritte ausführen soll.

Programmausschnitt (13):

```
3990 REM *****************************
4000 REM *          Computerzug      *
4010 REM *****************************
4020 :
4030 fu = 2
4040 ro = 0
4050 fm = 0
4060 WINDOW SWAP 1,0
4070 LOCATE 1,1
4080 PRINT "Ich"
4090 PRINT "werte:"
4100 PRINT "__-__"
4110 FOR pm=11 TO 88
4120 IF f(pm) <= 0 GOTO 4160
4130 IF f(pm) = 10 GOTO 4160
4140 m = f(pm)
4150 GOSUB 7600
4160 NEXT pm
4170 IF f(58) <> 8 GOTO 4270
4180 pm(1) = 58
4190 FOR a=38 TO 78 STEP 40
4200 pm(2) = a
4210 GOSUB 7100
4220 IF fl = 1 GOTO 4260
4230 fm = fm + 1
4240 fm(fm, 1) = pm(1)
4250 fm(fm, 2) = pm(2)
4260 NEXT a
```

Erläuterungen zum nebenstehenden Programmausschnitt (13):

Zu Beginn des Funktionsmoduls Computerzug setzen wir die Variable **fu** (**fu**nction) auf den Wert 2. **fu** dient zur Steuerung des Programmlaufs in der Subroutine **Feldbestimmung**, in der die legalen Züge einer ausgewählten Figur (**m**) auf einem Feld **pm** erzeugt werden. fu wird hier deshalb auf 2 gesetzt, um zu erreichen, daß erzeugte Computerzüge in der Variablen fm(..,..) gespeichert werden. Die anderen Werte für fu - nämlich 1, 3 und 4 - erläutern wir an späterer Stelle.

Der Merker ro, der hier auf 0 gesetzt wir, erhält nur dann einen Wert <> 0, wenn der Rechner einen Rochadezug erzeugt hat.

Mit Hilfe der (nicht indizierten!) Variablen fm zählt das System die Anzahl der erzeugten Züge.

Nachdem wir mit der Anweisung in Zeile 4060 den Bildschirmbereich WINDOW#1 aktiviert haben, geben wir hier in den Zeilen 4070 bis 4100 folgenden Text aus:

```
Ich
werte:

__-__
```

In den Zeilen 4110 bis 4160 berechnet der Computer durch Aufruf der Subroutine Feldbestimmung (ab Zeile 7600) alle legalen Züge (unter Mißachtung einer möglichen Königsbedrohung), mit Ausnahme der Rochaden, die in den Zeilen 4170 bis 4260 ermittelt werden.

Programmausschnitt (14):

```
4270 FOR a=1 TO fm
4280 pm(1) = fm(a, 1)
4290 pm(2) = fm(a, 2)
4300 GOSUB 5100
4310 GOSUB 5400
4320 GOSUB 6700
4330 GOSUB 7400
4340 v = -9990
4350 IF fl = 0 THEN GOSUB 5800
4360 fm(a, 3) = v
4370 GOSUB 6900
4380 NEXT a
```

Erläuterungen zum nebenstehenden Programmausschnitt (14):

Anschließend bewertet das System in den Zeilen 4270 bis 4380 die vorher berechneten Züge in einer FOR ... NEXT- Schleife.

Die Bewertung im einzelnen:

Zeilen 4280 - 4290: Zuweisung der Start- und Zielkoordinate des zu bewertenden Zuges nach pm(1) und pm(2),

Zeile 4300: Aufruf der Subroutine "Zugparameter einstellen"; hier untersuchen wir, ob es sich um einen "en-passant-Schlag" oder um eine Rochade handelt, und berechnen dementsprechend pm(3) und pm(4).

Zeilen 4310 - 4330: Aufruf der Subroutinen
Zugausgabe in WINDOW#1 (ab Zeile 5400),
Interne Zugausführung (ab Zeile 6700),
König im Schach ? (ab Zeile 7400).

Zeilen 4340 - 4360: In diesen Zeilen findet die eigentliche Bewertung des Zuges statt. Zu diesem Zweck weisen wir zunächst der Variablen **v** (**v**alue) den niedrigsten Stellungswert zu, den wir mit -9990 festgelegt haben. Wenn der "Fehler-Merker" fl den Wert 0 hat - das ist dann der Fall, wenn in der Subroutine ab Zeile 7400 keine Königsbedrohung festgestellt wurde - rufen wir die in Zeile 5800 beginnende Bewertungsfunktion auf, wo der Wert der durch den aktuellen Zug entstandenen Stellung berechnet und in v abgelegt wird. Abschließend weisen wir den Wert von v der Variablen fm(a, 3) zu.

Zeile 4370: Aufruf der Subroutine
Interne Zugzurücknahme

Programmausschnitt (15):

```
4390 GOSUB 5600

4400 IF fm(1,3) > -9990 GOTO 4440

4410 GOSUB 7400

4420 IF fl = 1 THEN ma = 2 ELSE ma = 1

4430 GOTO 5060

4440 FOR a=fm TO 1 STEP -1

4450 IF fm(a,3) = -9990 THEN fm = a - 1

4460 NEXT a

4470 IF mo = 1 GOTO 5010
```

Erläuterungen zum nebenstehenden Programmausschnitt (15):

In Zeile 4390 erfolgt der Aufruf einer Subroutine, mit der wir die bewerteten Züge nach den ermittelten Stellungswerten sortieren, um dann den in fm(1,3) gespeicherten Wert des besten Zuges darauf zu untersuchen, ob er -9990 beträgt. Ist dies der Fall, steht der König des Rechners mit Sicherheit im Patt, weil er keinen Zug mehr ausführen kann, ohne in eine Bedrohungssituation zu geraten (vgl. hierzu auch unsere Ausführungen auf Seite 2-49!). In einem solchen Patt-Fall wird durch Aufruf einer in Zeile 7400 beginnenden Subroutine (Zeile 4410) untersucht, ob der König im Schach steht. Ist dies der Fall, erhält fl den Wert 1, mit der Folge, daß ma auf 2 (= matt) gesetzt wird. Andernfalls weisen wir ma den Wert 1 (= patt) zu. Anschließend verzweigen wir zum Ende des Funktionsmoduls.

In der FOR ... NEXT-Schleife der Zeilen 4440 bis 4460 reduzieren wir den Zähler fm um die Anzahl der königsbedrohenden Züge.

Wenn sich der Anwender zu Beginn des Programmlaufs für die Spielstärke 1 (mo = 1) entschieden hat, bewirkt die Abfrage in Zeile 4470 das Überspringen des ab der folgenden Seite beschriebenen Programmausschnitts (Zeilen 4480 bis 5000), wodurch eine beträchtliche Verkürzung der Antwortzeiten zu Lasten der Spielqualität des Rechners herbeigeführt wird.

Programmausschnitt (16):

```
4480 v0 = -9999
4490 FOR a=1 TO fm
4500 fm(a, 3) = 9999
4510 pm(1) = fm(a, 1)
4520 pm(2) = fm(a, 2)
4530 st = 1
4540 GOSUB 5100
4550 GOSUB 5400
4560 GOSUB 6700
4570 FOR b=1 TO 4
4580 pm( b + 4 ) = pm(b)
4590 NEXT b
4600 fh = f0 : sh = sm : sm = 0
4610 pc = bm
4620 st = -1
4630 WHILE pc <= bm + 99
4640 mm = 0
4650 pm = pc MOD 100
4660 IF f(pm) >= 0 GOTO 4910
4670 m = -f(pm)
4680 fu = 3
4690 GOSUB 7600
4700 b = 1
4710 WHILE b <= mm
4720 pm(1) = pc MOD 100
4730 pm(2) = mm(b)
4740 GOSUB 5100
4750 GOSUB 6700
4760 GOSUB 7400
4770 IF f1 = 1 GOTO 4850
4780 GOSUB 5800
4790 IF v > v0 GOTO 4820
4800 bm = pc MOD 100
4810 b = mm : pc = 200
4820 IF fm(a, 3) = -9999 GOTO 4840
4830 IF v >= fm(a, 3) GOTO 4850
4840 fm(a, 3) = v
4850 GOSUB 6900
4860 IF f1 = 0 GOTO 4890
4870 GOSUB 7400
4880 IF f1 = 0 THEN fm(a, 3) = -9999
4890 b = b + 1
4900 WEND
4910 IF pc=88 THEN pc=111 ELSE pc=pc+1
4920 WEND
4930 IF v0 < fm(a,3) THEN v0 = fm(a,3)
4940 FOR b=1 TO 4
4950 pm(b) = pm( b + 4 )
4960 NEXT b
4970 f0 = fh : sm = sh : st = 1
4980 GOSUB 6900
4990 NEXT a
5000 GOSUB 5600
```

Erläuterungen zum nebenstehenden Programmausschnitt (16):

In dem Programmabschnitt der Zeilen 4480 bis 5000, der die Erhöhung der Spielqualität (= **Spielstärke 2**) bewirkt, bedienen wir uns des "**Minimax-Prinzips**" unter Verwendung der "**Alpha-Beta-Methode**".

Das uneingeschränkte **Minimax-Prinzip** beinhaltet folgende Bewertungsstrategie der vom Rechner gespeicherten Züge, die wir fortan - der üblichen Terminologie folgend - als Halbzüge bezeichnen wollen: Zu jedem Halbzug des Rechners erzeugt dieser sämtliche Gegenzüge (Halbzüge), die der Spieler ausführen könnte, und nimmt dazu eine Stellungsbewertung vor. Da sich der Wert einer Stellung - für Erklärungszwecke zunächst sehr **vereinfacht** - aus der Summe der positiven Figurenwerte des Computers und der negativen Figurenwerte des Spielers ergibt, würde der Spieler mit jenem Halbzug antworten, der den niedrigsten Stellungswert ergibt. Der Computer "verhält" sich dann "vernünftig", wenn er zu all seinen Halbzügen den bewertungsminimalen Gegenzug ermittelt und dann jenen Halbzug tätigt, der den maximalen Minimalwert des Gegenzugs zur Folge hat (= Minimax-Prinzip).

Die oben geschilderte Methode würde bereits auf der Ebene des 2. Halbzugs zu einer im Hinblick auf das Zeitverhalten des Schachprogramms unverhältnismäßig hohen Anzahl zu registrierender und zu untersuchender Stellungswerte führen. Auf der Suche nach einer Verkürzungsmöglichkeit entdeckte man im Jahre 1959 die **Alpha-Beta-Methode**:

Für ihren Einsatz ist es empfehlenswert (jedoch nicht zwingend erforderlich), daß die möglichen Halbzüge des Rechners nach absteigender Wertigkeit sortiert sind. Man setzt die Untersuchung nun dergestalt fort, daß man zunächst zum besten Halbzug des Rechners sämtliche möglichen Gegen-Halbzüge des Spielers ermittelt und hiervon jenen registriert, der eine minimale

Programmausschnitt (16):

```
4480 v0 = -9999
4490 FOR a=1 TO fm
4500 fm(a, 3) = 9999
4510 pm(1) = fm(a, 1)
4520 pm(2) = fm(a, 2)
4530 st = 1
4540 GOSUB 5100
4550 GOSUB 5400
4560 GOSUB 6700
4570 FOR b=1 TO 4
4580 pm( b + 4 ) = pm(b)
4590 NEXT b
4600 fh = f0 : sh = sm : sm = 0
4610 pc = bm
4620 st = -1
4630 WHILE pc <= bm + 99
4640 mm = 0
4650 pm = pc MOD 100
4660 IF f(pm) >= 0 GOTO 4910
4670 m = -f(pm)
4680 fu = 3
4690 GOSUB 7600
4700 b = 1
4710 WHILE b <= mm
4720 pm(1) = pc MOD 100
4730 pm(2) = mm(b)
4740 GOSUB 5100
4750 GOSUB 6700
4760 GOSUB 7400
4770 IF fl = 1 GOTO 4850
4780 GOSUB 5800
4790 IF v > v0 GOTO 4820
4800 bm = pc MOD 100
4810 b = mm : pc = 200
4820 IF fm(a, 3) = -9999 GOTO 4840
4830 IF v >= fm(a, 3) GOTO 4850
4840 fm(a, 3) = v
4850 GOSUB 6900
4860 IF fl = 0 GOTO 4890
4870 GOSUB 7400
4880 IF fl = 0 THEN fm(a, 3) = -9999
4890 b = b + 1
4900 WEND
4910 IF pc=88 THEN pc=111 ELSE pc=pc+1
4920 WEND
4930 IF v0 < fm(a,3) THEN v0 = fm(a,3)
4940 FOR b=1 TO 4
4950 pm(b) = pm( b + 4 )
4960 NEXT b
4970 f0 = fh : sm = sh : st = 1
4980 GOSUB 6900
4990 NEXT a
5000 GOSUB 5600
```

Erläuterungen zum nebenstehenden Programmausschnitt (16) / Forts.:

Bewertungszahl (= Optimum aus der "fiktiven Sicht des Spielers") ergibt. Für den nächstbesten Halbzug des Computers brauchen die möglichen Gegenzüge des Spielers nur nur solange untersucht zu werden, wie die Bewertungszahlen höher liegen als der für den zunächst besten Halbzug des Computers ermittelte Spieler-Minimalwert. Sobald jetzt eine Bewertungszahl ermittelt wird, die niedriger ist, kann die Untersuchung der möglichen Spieler-Reaktionen auf den Computer-Halbzug abgebrochen werden: In Anlehnung an die englische Sprache bezeichnet man diese Vorgehensweise als "Alpha-Pruning" (= "Alpha-Schnitt"). Wird allerdings zu einem Computer-Halbzug in der dazugehörigen "Liste" der "fiktiven Spieler-Halbzüge" ein Minimalwert gefunden, der höher ist als der bisherige, so wird dieser statt des bisherigen gespeichert und bestimmt somit einen zunächst niedriger bewerteten Computer-Halbzug als den besseren.

Wenn nach der verkürzten Analyse der Spieler-Halbzüge mittels "Alpha-Pruning" die Untersuchung für die möglichen Computer-Halbzüge fortgesetzt werden würde, so müßte dann in analoger Weise ein möglicher Halbzug sofort verworfen werden, wenn er einen Stellungswert zur Folge hat, der niedriger ist als der jeweilige Maximalwert (= "Beta-Pruning").

In unserem Programm beschränken wir uns - vor allem im Hinblick auf die verhältnismäßig langsame Rechengeschwindigkeit eines in der Interpreter-Sprache BASIC verfaßten Programms - auf das "Alpha-Pruning". Auch wollen wir keine zusätzlichen Selektionsstrategien behandeln, um die Logik des zu erläuternden Programms möglichst einfach zu halten. Nach der Einarbeitung in die Thematik wird es dem Leser sicherlich nicht schwer fallen, eigene Ideen kurzfristig zu realisieren, um deren Vor- und Nachteile in entsprechenden Tests zu anlysieren.

Programmausschnitt (16):

```
4480 v0 = -9999
4490 FOR a=1 TO fm
4500 fm(a, 3) = 9999
4510 pm(1) = fm(a, 1)
4520 pm(2) = fm(a, 2)
4530 st = 1
4540 GOSUB 5100
4550 GOSUB 5400
4560 GOSUB 6700
4570 FOR b=1 TO 4
4580 pm( b + 4 ) = pm(b)
4590 NEXT b
4600 fh = f0 : sh = sm : sm = 0
4610 pc = bm
4620 st = -1
4630 WHILE pc <= bm + 99
4640 mm = 0
4650 pm = pc MOD 100
4660 IF f(pm) >= 0 GOTO 4910
4670 m = -f(pm)
4680 fu = 3
4690 GOSUB 7600
4700 b = 1
4710 WHILE b <= mm
4720 pm(1) = pc MOD 100
4730 pm(2) = mm(b)
4740 GOSUB 5100
4750 GOSUB 6700
4760 GOSUB 7400
4770 IF f1 = 1 GOTO 4850
4780 GOSUB 5800
4790 IF v > v0 GOTO 4820
4800 bm = pc MOD 100
4810 b = mm : pc = 200
4820 IF fm(a, 3) = -9999 GOTO 4840
4830 IF v >= fm(a, 3) GOTO 4850
4840 fm(a, 3) = v
4850 GOSUB 6900
4860 IF f1 = 0 GOTO 4890
4870 GOSUB 7400
4880 IF f1 = 0 THEN fm(a, 3) = -9999
4890 b = b + 1
4900 WEND
4910 IF pc=88 THEN pc=111 ELSE pc=pc+1
4920 WEND
4930 IF v0 < fm(a,3) THEN v0 = fm(a,3)
4940 FOR b=1 TO 4
4950 pm(b) = pm( b + 4 )
4960 NEXT b
4970 f0 = fh : sm = sh : st = 1
4980 GOSUB 6900
4990 NEXT a
5000 GOSUB 5600
```

Erläuterungen zum nebenstehenden Programmausschnitt (16) / Forts.:

Die Befehle im einzelnen:

Zeile 4480: Wir weisen der Variablen v0 einen vorläufigen Minimax-Wert für das "Alpha-Pruning" zu, um damit zu gewährleisten, daß zum ersten untersuchten Computer-Halbzug alle möglichen Spieler-Halbzüge berechnet und bewertet werden, denn mit dem vorläufigen Minimax-Wert von -9999 ist sichergestellt, daß alle Spieler-Stellungswerte zum ersten Computer-Halbzug höher sind.

Zeile 4490: Hier beginnt eine FOR ... NEXT-Schleife, die in Zeile 4490 endet. In ihr führen wir für alle Werte von 1 bis fm (= Anzahl der zulässigen Computer-Halbzüge) jeweils einen Computer-Halbzug und eine durch das "Alpha-Pruning" beschränkte Anzahl von Spieler-Halbzügen durch.

Zeile 4500: Wir legen hier einen vorläufigen höchstmöglichen Stellungswert von 9999 fest, weil so bei der Suche nach dem niedrigstmöglichen Stellungswert auf einfache Weise gewährleistet ist, daß bereits der erste Wert für alle weiteren Vergleiche übernommen wird.

Zeilen 4510 - 4560: Nachdem wir die Start- und Zielkoordinate des Computer-Halbzugs in pm(1) und pm(2) abgelegt und die Status-Variable st auf 1 gesetzt haben, werden drei Subroutinen aufgerufen, um
- die Zugparameter einzustellen,
- die Zugausgabe in WINDOW#1 vorzunehmen und
- den Zug intern auszuführen.

Programmausschnitt (16):

```
4480 v0 = -9999
4490 FOR a=1 TO fm
4500 fm(a, 3) = 9999
4510 pm(1) = fm(a, 1)
4520 pm(2) = fm(a, 2)
4530 st = 1
4540 GOSUB 5100
4550 GOSUB 5400
4560 GOSUB 6700
4570 FOR b=1 TO 4
4580 pm( b + 4 ) = pm(b)
4590 NEXT b
4600 fh = f0 : sh = sm : sm = 0
4610 pc = bm
4620 st = -1
4630 WHILE pc <= bm + 99
4640 mm = 0
4650 pm = pc MOD 100
4660 IF f(pm) >= 0 GOTO 4910
4670 m = -f(pm)
4680 fu = 3
4690 GOSUB 7600
4700 b = 1
4710 WHILE b <= mm
4720 pm(1) = pc MOD 100
4730 pm(2) = mm(b)
4740 GOSUB 5100
4750 GOSUB 6700
4760 GOSUB 7400
4770 IF f1 = 1 GOTO 4850
4780 GOSUB 5800
4790 IF v > v0 GOTO 4820
4800 bm = pc MOD 100
4810 b = mm : pc = 200
4820 IF fm(a, 3) = -9999 GOTO 4840
4830 IF v >= fm(a, 3) GOTO 4850
4840 fm(a, 3) = v
4850 GOSUB 6900
4860 IF f1 = 0 GOTO 4890
4870 GOSUB 7400
4880 IF f1 = 0 THEN fm(a, 3) = -9999
4890 b = b + 1
4900 WEND
4910 IF pc=88 THEN pc=111 ELSE pc=pc+1
4920 WEND
4930 IF v0 < fm(a,3) THEN v0 = fm(a,3)
4940 FOR b=1 TO 4
4950 pm(b) = pm( b + 4 )
4960 NEXT b
4970 f0 = fh : sm = sh : st = 1
4980 GOSUB 6900
4990 NEXT a
5000 GOSUB 5600
```

Erläuterungen zum nebenstehenden Programmausschnitt (16) / Forts.:

Zeilen 4570 - 4600: Hier sichern wir die Werte der Variablen pm(1) bis pm(4) nach pm(5) bis pm(8) und f0 - diese Variable enthält die vorherige Feldbelegung des Zielkoordinatenfeldes - nach fh sowie sm nach sh (swap help), weil pm(1) bis pm(4), f0 und sm vor Zurücknahme des Computer-Halbzugs noch für den Spieler-Halbzug benötigt werden. Dann erhält sm wieder seinen Ausgangswert 0.

Zeile 4610: Wir weisen **pc** (position counter) den in der Variablen **bm** (best move) enthaltenen Anfangswert zu. Sie wurde im Funktionsmodul **Initialisierung** (vgl. Seite 2-30!) auf den Wert 11 gesetzt. Später speichern wir in ihr jene Feldposition, bei der der letzte "Alpha-Schnitt" (vgl. Seite 2-55!) durchgeführt wurde. Weil sehr häufig bei der nächsten Untersuchung der möglichen Spieler-Halbzüge (aufgrund des nächsten Computer-Halbzugs) das "Alpha-Pruning" an der gleichen Stelle einsetzt, erreichen wir so im Regelfall eine Beschleunigung der Bearbeitung des zweiten Halbzugs (= Spielstärke 2). Beispielsweise erfolgt so die Zählung für pm von 41 bis 88 und dann von 11 bis 40, statt von 11 bis 88.

Zeile 4620: Die Status-Variable st erhält den Wert -1, weil jetzt Spielerzüge bearbeitet werden sollen. In der

ab
Zeile 4630 beginnenden WHILE ... WEND-Schleife werden Figuren des Spielers auf dem Schachbrett gesucht (Zeilen 4640 - 4670), durch Aufruf

Programmausschnitt (16):

```
4480 v0 = -9999
4490 FOR a=1 TO fm
4500 fm(a, 3) = 9999
4510 pm(1) = fm(a, 1)
4520 pm(2) = fm(a, 2)
4530 st = 1
4540 GOSUB 5100
4550 GOSUB 5400
4560 GOSUB 6700
4570 FOR b=1 TO 4
4580 pm( b + 4 ) = pm(b)
4590 NEXT b
4600 fh = f0 : sh = sm : sm = 0
4610 pc = bm
4620 st = -1
4630 WHILE pc <= bm + 99
4640 mm = 0
4650 pm = pc MOD 100
4660 IF f(pm) >= 0 GOTO 4910
4670 m = -f(pm)
4680 fu = 3
4690 GOSUB 7600
4700 b = 1
4710 WHILE b <= mm
4720 pm(1) = pc MOD 100
4730 pm(2) = mm(b)
4740 GOSUB 5100
4750 GOSUB 6700
4760 GOSUB 7400
4770 IF fl = 1 GOTO 4850
4780 GOSUB 5800
4790 IF v > v0 GOTO 4820
4800 bm = pc MOD 100
4810 b = mm : pc = 200
4820 IF fm(a, 3) = -9999 GOTO 4840
4830 IF v >= fm(a, 3) GOTO 4850
4840 fm(a, 3) = v
4850 GOSUB 6900
4860 IF fl = 0 GOTO 4890
4870 GOSUB 7400
4880 IF fl = 0 THEN fm(a, 3) = -9999
4890 b = b + 1
4900 WEND
4910 IF pc=88 THEN pc=111 ELSE pc=pc+1
4920 WEND
4930 IF v0 < fm(a,3) THEN v0 = fm(a,3)
4940 FOR b=1 TO 4
4950 pm(b) = pm( b + 4 )
4960 NEXT b
4970 f0 = fh : sm = sh : st = 1
4980 GOSUB 6900
4990 NEXT a
5000 GOSUB 5600
```

Erläuterungen zum nebenstehenden Programmausschnitt (16) / Forts.:

des Unterprogramms Feldbestimmung in Zeile 4690 - abhängig von fu=3 - alle möglichen Züge einer gefundenen Figur erzeugt und nach dem bereits erläuterten Prinzip des "Alpha-Pruning" in der inneren WHILE ... WEND-Schleife der

Zeilen 4710 - 4900 bewertet. - In

Zeile 4930 aktualisieren wir ggf. den Minimax-Wert. Nachdem wir mit den Anweisungen der

Zeilen 4940 - 4970 die Werte der Variablen pm(5) bis pm(8) wieder nach pm(1) bis pm(4), fh nach f0, sh nach sm zurückübertragen und st wieder auf 1 gesetzt haben, rufen wir in

Zeile 4980 ein Unterprogramm auf, mit dem wir den Computer-Halbzug rückgängig machen.

In

Zeile 4990 finden wir den letzten Befehl der FOR ... NEXT-Schleife, die mit der Anweisung in Zeile 4490 begann. Es endet somit die Bewertung eines Computer-Halbzugs und mit dem Rücksprung zum Schleifenanfang kann die Bewertung eines weiteren Computer-Halbzugs eingeleitet werden.

Zeile 5000: Sobald alle Computer-Halbzüge bewertet worden sind, erfolgt in einer Subroutine ab Zeile 5600 eine Sortierung nach den Stellungswerten.

Programmausschnitt (17):

```
5010 pm(1) = fm(1, 1)
5020 pm(2) = fm(1, 2)
5030 GOSUB 5100
5040 IF fm(1, 3) =  9999 THEN ma = 2
5050 IF fm(1, 3) = -9999 THEN ma = 1
5060 WINDOW SWAP 1,0
5070 st = 1
5080 RETURN
5090 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (17):

	Nachdem wir in den
Zeilen 5010 - 5020	die Start- und Zielkoordinate des besten Computer-Halbzugs nach pm(1) und pm(2) übertragen haben, rufen wir in
Zeile 5030	eine Subroutine (ab Zeile 5100) auf, in der wir die zusätzlichen Parameter für pm(3) und pm(4) berechnen. Anschließend untersuchen wir in den
Zeilen 5040 - 5050,	ob der soeben berechnete (aber noch nicht endgültig ausgeführte) beste Zug den Wert 9999 (= der Spieler ist schachmatt und der Merker ma erhält den Wert 2) oder -9999 (= der Spieler ist im Patt und der Merker ma erhält den Wert 1) hat. Mit dem Befehl der
Zeile 5060	reaktivieren wir den Gesamtbildschirm als Ausgabeeinheit, weisen in
Zeile 5070	der Status-Variablen st den Wert 1 zu, um damit sicherzustellen, daß im Funktionsmodul Zugausgabe der auszuführende Halbzug als ein solcher des Computers behandelt wird. Mit der Anweisung in
Zeile 5080	kehren wir schließlich in die Steuerleiste zurück.

Programmausschnitt (18):

```
5100 REM *** Zugparameter einstellen **
5110 :
5120 IF f(pm(2))=0 THEN f1=0 ELSE f1=1
5130 pm(3) = 0
5140 pm(4) = 0
5150 IF f( pm(1) ) <> 2 * st GOTO 5240
5160 c = 4.5 - st * 0.5
5170 IF pm(1) MOD 10 <> c GOTO 5240
5180 IF pm(1)-pm(2) = 11*st GOTO 5200
5190 IF pm(2)-pm(1) <> 9*st GOTO 5290
5200 pm(3) = pm(2) + st
5210 f1 = 1
5220 GOTO 5290
5230 :
5240 IF f( pm(1) ) <> 8 * st GOTO 5290
5250 ro = ( pm(2) - pm(1) ) / 2
5260 IF ABS(ro) <> 10 GOTO 5290
5270 pm(3) = 49.5 + 3.5 * ( st + ro )
5280 pm(4) = pm(1) + ro
5290 RETURN
5300 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (18):

In der nebenstehenden Subroutine "Zugparameter einstellen" untersuchen wir, ob die in pm(1) und pm(2) enthaltenen Koordinaten für das Start- und Zielfeld einen "en-passant-Schlag" (Zeilen 5150 - 5200) oder eine Rochade (Zeilen 5240 - 5280) darstellen, und berechnen ggf. die Werte für die Variablen pm(3) und pm(4). (Vgl. hierzu auch unsere Ausführungen auf Seite 2-17!)

Darüber hinaus wird der Merker fl auf 1 gesetzt (Zeilen 5120 und 5210), wenn durch den in pm(1) bis pm(4) gespeicherten Halbzug eine Figur geschlagen wird.

Programmausschnitt (19):

```
5400 REM *** Zugausgabe in WINDOW#1 ***
5410 :
5420 LOCATE 1,4
5430 FOR aa = 1 TO 2
5440 PRINT CHR$( INT( pm(aa)/10 ) +64);
5450 PRINT CHR$( pm(aa) MOD 10 + 48 );
5460 IF aa = 2 GOTO 5490
5470 IF fl = 0 THEN PRINT "-";
5480 IF fl = 1 THEN PRINT "x";
5490 NEXT aa
5500 RETURN
5510 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (19):

In der nebenstehenden Subroutine geben wir die in pm(1) und pm(2) gespeicherte Start- und Zielkoordinate eines Halbzuges in der üblichen Schachnotation aus.

Die Umrechnung von der internen Notation in die übliche Schachnotation, die wir anhand eines Beispiels erläutern wollen, erfolgt durch die Formeln in den Ausgabebefehlen der Zeilen 5440 und 5450.

Beispiel:

Umwandlung und Ausgabe der internen Koordinate **52**, die dem Feld **E2** des üblichen Schachbretts entspricht:

1. Schritt:	pm(aa)/10 also 52/10	=	5.2
2. Schritt:	INT(5.2)	=	5
3. Schritt:	5 + 64	=	69
4. Schritt:	CHR$(69)	=	**E**
5. Schritt:	Ausgabe von "E"		
6. Schritt:	pm(aa) MOD 10 also 52 MOD 10	=	2
7. Schritt:	2 + 48	=	50
8. Schritt:	CHR$(50)	=	**2**
9. Schritt:	Ausgabe von "2"		

Programmausschnitt (20):

```
5600 REM *** Sortieren von fm(a,b) ***
5610 :
5620 LOCATE 1,4
5630 PRINT "__-__"
5640 FOR a=1 TO fm - 1
5650 FOR b=a + 1 TO fm
5660 IF fm(a, 3) >= fm(b, 3) GOTO 5720
5670 FOR c=1 TO 3
5680 f0 = fm(a, c)
5690 fm(a, c) = fm(b, c)
5700 fm(b, c) = f0
5710 NEXT c
5720 NEXT b, a
5730 RETURN
5740 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (20):

In der aufgeführten Routine, die aus dem Programmausschnitt (15) - vgl. hierzu Seite 2-51 - und aus dem Programmausschnitt (16) - vgl. hierzu Seite 2-61 - aufgerufen wird, sortieren wir die in fm(..,1) und fm(..,2) gespeicherten Start- und Zielkoordinaten des Computer-Halbzugs in absteigender Sortierfolge der Stellungswerte, die in fm(..,3) abgelegt sind.

Programmausschnitt (21):

```
5800 REM *** Bewertungsfunktion ***
5810 :
5820 fu = 4
5830 v = 0
5840 sg = SGN(f0)
5850 v = v - 2 * vm( f0 * sg ) * sg
5860 sg = SGN(fh)
5870 v = v - 2 * vm( fh * sg ) * sg
5880 :
5890 FOR pm=11 TO 88
5900 IF f(pm) = 0 GOTO 5950
5910 IF f(pm) = 10 GOTO 5950
5920 sg = SGN( f(pm) )
5930 m = ABS( f(pm) )
5940 GOSUB 7600
5950 NEXT pm
5960 RETURN
5970 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (21):

Die Subroutine Bewertungsfunktion ist im gewissen Sinne eines der Kernstücke des Schachprogramms, weil durch Änderungen in diesem Abschnitt ein anderes Spielverhalten des Computers herbeigeführt werden kann.

Um dem Leser einen möglichst großen Spielraum für eigene Bewertungskriterien einzuräumen, haben wir uns hier um besondere Einfachheit und Durchsichtigkeit bemüht:

Zu Beginn setzen wir die Variable fu auf 4, um dadurch den Programmlauf in der Subroutine Feldbestimmung zu steuern, die in Zeile 5940 aufgerufen wird (vgl. hierzu auch unsere Ausführungen auf Seite 2-47!). Anschließend erhält die Variable v, welche die jeweilige Bewertungszahl aufzunehmen hat, den Wert 0, um so den Wert eines vorangegangenen Bewertungsdurchlaufs zu löschen.

Die Bewertung im einzelnen:

Zeilen 5840 - 5870:

In unseren Ausführungen zum Programmausschnitt (16) auf Seite 2-53 hatten wir für Erläuterungszwecke zunächst **vereinfacht** unterstellt, daß sich der Wert einer Stellung aus der Summe der positiven Figurenwerte des Computers und der negativen Figurenwerte des Spielers ergibt. Wir differenzieren diese Aussage jetzt dahingehend, daß die Summe der Figurenwerte einer Stellung als Konstante zu betrachten ist, und sich daher der Wert einer Stellung lediglich aus dem Wert der zunächst nur für interne Berechnungszwecke geschlagenen Figur (bei Spielstärke 2: Figur<u>en</u>) ergibt.

Beispiel:

Es wurde ein Turm des Spielers (Wert: - 25) geschlagen. Weil man die negativen Werte der geschlagenen Figur des Spielers bei der Stellungswertermittlung abziehen (= addieren) muß, damit sich das

Programmausschnitt (21):

```
5800 REM *** Bewertungsfunktion ***
5810 :
5820 fu = 4
5830 v = 0
5840 sg = SGN(f0)
5850 v = v - 2 * vm( f0 * sg ) * sg
5860 sg = SGN(fh)
5870 v = v - 2 * vm( fh * sg ) * sg
5880 :
5890 FOR pm=11 TO 88
5900 IF f(pm) = 0 GOTO 5950
5910 IF f(pm) = 10 GOTO 5950
5920 sg = SGN( f(pm) )
5930 m = ABS( f(pm) )
5940 GOSUB 7600
5950 NEXT pm
5960 RETURN
5970 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (21) / Forts.:

Schlagen einer Figur des Spielers positiv auswirkt, und jene des Computers (im 2. Halbzug) effektiv abzuziehen sind, damit sich das Schlagen einer Computer-Figur für den Rechner negativ auswirkt, müssen wir zunächst das Vorzeichen der geschlagenen Figur ermitteln (Zeile 5840); **sg** (**sign**) erhält hier also den Wert -1. Mit der Formel in Zeile 5850 wird der Wert der geschlagenen Figur folgendermaßen verrechnet:

1. Schritt: **f0** * **sg**
   (Ziffer der zu repräsentierenden Figur/hier: -5) -1
   = **5**

2. Schritt: **vm(5)**
   (enthält den Absolutwert der Figur, nämlich **25**)

3. Schritt: **25** * **sg**
   (Absolutwert der Figur) (Vorzeichen)
   = **-25** (= Realwert der Figur, d. h. differenziert nach Spieler bzw. Computer)

4. Schritt: Da der Wert einer Figur, die geschlagen wurde, höher angesetzt werden soll als jene, die geschlagen werden könnte, verdoppeln wir den Realwert der Figur:

   2 * **-25**
   = **-50** (= gewichteter Realwert der Figur)

5. Schritt: **v** - **-50**
   = **v + 50**

Daraus folgt, daß der alte Inhalt von v um 50 erhöht wird, da der Computer den Turm des Spielers in dem untersuchten Halbzug geschlagen hat.

Programmausschnitt (21):

```
5800 REM *** Bewertungsfunktion ***
5810 :
5820 fu = 4
5830 v = 0
5840 sg = SGN(f0)
5850 v = v - 2 * vm( f0 * sg ) * sg
5860 sg = SGN(fh)
5870 v = v - 2 * vm( fh * sg ) * sg
5880 :
5890 FOR pm=11 TO 88
5900 IF f(pm) = 0 GOTO 5950
5910 IF f(pm) = 10 GOTO 5950
5920 sg = SGN( f(pm) )
5930 m = ABS( f(pm) )
5940 GOSUB 7600
5950 NEXT pm
5960 RETURN
5970 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (21) / Forts.:

Mit den Befehlen der Zeilen 5860 bis 5870 wiederholen wir die geschilderte Prozedur für den 2. Halbzug des Spielers. Bei Spielstärke 1 sind diese Befehle zwar eigentlich überflüssig. Da v aber für alle Stellungsbewertungen in gleicher Weise um einen bestimmten Wert, abhängig von fh, verändert wird, konnten wir darauf verzichten, die Befehle mit Hilfe eines Merkers zu überspringen.

Die sich anschließende FOR ... NEXT-Schleife (Zeilen 5890 - 5950) **ergänzt** das oben erläuterte Bewertungsverfahren um einen spielstrategischen Aspekt, und zwar ermitteln wir durch Aufruf der Subroutine Feldbestimmung (in Zeile 5940) für **jede** Figur alle erreichbaren Felder und vergeben hier für ein "beherrschtes", leeres Feld einen Punkt und für eine schlagbare Figur ihren Figurenwert (vgl. Seite 2-21!). Dabei ergeben die Berechnungen für die Figuren des Computers positive und für jene des Spielers negative Zahlen, die zur Bewertungsvariablen v addiert werden.

Programmausschnitt (22):

```
5990 REM ******************************
6000 REM *          Zugausgabe          *
6010 REM ******************************
6020 :
6030 GOSUB 6700
6040 IF fm(1, 3) = -9990 GOTO 6430
6050 IF pm(4) = 0 THEN c = 3 ELSE c = 4
6060 IF pm(3) = 0 THEN c = 2
6070 FOR a=1 TO c
6080 pf = pm(a)
6090 GOSUB 6500
6100 NEXT a
6110 FOR a = 5 TO 8 STEP 3
6120 IF f( pm(2) ) <> a * st GOTO 6140
6130 f( pm(2) ) = a * st + st
6140 NEXT a
6150 c = 4.5 - st * 0.5
6160 FOR a=10 + c TO 80 + c STEP 10
6170 IF f(a) <> -st GOTO 6190
6180 f(a) = f(a) - st
6190 NEXT a
6200 c = c + st * 2
6210 IF f( pm(2) ) <> st GOTO 6250
6220 IF pm(2) MOD 10 <> c GOTO 6250
6230 f( pm(2) ) = f( pm(2) ) + st
6240 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (22):

Das Funktionsmodul **Zugausgabe** dient der endgültigen Zugausführung, und zwar sowohl für interne Zwecke in den Variablen f(..) durch Aufruf der Subroutine "Interne Zugausführung" in Zeile 6030 als auch auf dem Bildschirm. Nur wenn der Computer matt oder patt ist (fm(1, 3) = -9990), wird kein Zug mehr ausgeführt (Zeile 6040). - Für die Ausgabe auf dem Bildschirm unterstellen wir zunächst, daß durch den Halbzug 4 Felder (= Rochade) betroffen sind (Zeile 6050), um dann durch die Befehle der Zeilen 6050 und 6060 den Wert der Variablen c zu modifizieren, wenn ein "en-passant-Schlag" (c = 3) oder ein "üblicher Zug" (c = 2) vorliegt. In der FOR ... NEXT-Schleife der Zeilen 6070 bis 6100 werden dann die betroffenen Felder mit Hilfe der Subroutine ab Zeile 6500 neu belegt. Mittels der sich anschließenden FOR ... NEXT-Schleife der Zeilen 6110 bis 6140 setzen wir ggf. die Ziffer für den Turm von 5 auf 6 (bzw. von -5 auf -6) oder die des Königs von 8 auf 9 (bzw. von -8 auf -9); vgl. hierzu Seite 2-04!

Da ein "en-passant-Schlag" entsprechend den üblichen Schachregeln nur unmittelbar nach dem Doppelzug des gegnerischen Bauern durchgeführt werden darf, verwandeln wir die Ziffer eines um zwei Felder vorgerückten Bauern erst im anschließenden Halbzug von 1 in 2 (bzw. von -1 in -2), und stellen somit sicher, daß die Zulässigkeit eines "en-passant-Schritts" fehlerfrei festgestellt werden kann. Zu diesem Zweck ergibt sich durch den Befehl in Zeile 6150 für die Variable c entweder der Wert 4 oder 5 (= zu überprüfende Zeile, auf der ein Doppelschritt-Bauer des Spielers oder des Computers stehen kann). Mit diesem Wert der Variablen c überprüfen wir mit den Anweisungen der Zeilen 6160 bis 6190, ob der jeweilige Gegner im letzten Halbzug mit einem Bauern einen Doppelschritt ausgeführt hat, um ggf. die den Bauern repräsentierende Ziffer umzuwandeln.

Mit den Befehlen der Zeilen 6200 bis 6230 untersuchen wir für die Zeile 6 bzw. 3, ob es sich bei dem in Ausführung befindlichen Halbzug um den ersten Einzelschritt eines Bauern handelt, um ggf. dessen Ziffer von -1 in -2 (bzw. von 1 in 2) umzuwandeln.

Programmausschnitt (23):

```
6250 WINDOW SWAP 2,0
6260 IF pm(4) <> 0 GOTO 6380
6270 fl = 0
6280 IF f0 <> 0 OR pm(3) <> 0 THEN fl=1
6290 FOR a=1 TO 2
6300 PRINT CHR$( INT( pm(a)/10 ) + 64);
6310 PRINT CHR$( pm(a) MOD 10 + 48 );
6320 IF a = 2 GOTO 6350
6330 IF fl =  0 THEN PRINT "-";
6340 IF fl <> 0 THEN PRINT "x";
6350 NEXT a
6360 PRINT
6370 GOTO 6400
6380 IF ro = -10 THEN PRINT "0-0-0"
6390 IF ro =  10 THEN PRINT " 0-0 "
6400 IF st = 1 THEN PRINT
6410 WINDOW SWAP 2,0
6420 st = -st
6430 sm = 0
6440 IF ma > 0 THEN st = 0
6450 RETURN
6460 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (23):

In diesem Abschnitt schalten wir zuerst WINDOW#2 als Ausgabebereich ein, um dort den ausgeführten Zug in der üblichen Schachnotation auszugeben (vgl. hierzu Seite 2-31!).

Zu diesem Zweck überprüfen wir in Zeile 6260, ob pm(4) <> 0 (= Rochade) ist, um ggf. in den Zeilen 6380 bzw. 6390 in Abhängigkeit von ro die jeweiligen Zeichen auszugeben.

Nachdem wir in den Zeilen 6270 und 6280 den Merker fl auf 0 bzw. 1 gesetzt haben (= keine bzw. eine geschlagene Figur), geben wir in der folgenden FOR ... NEXT-Schleife (Zeilen 6290 - 6350) die in pm(1) und pm(2) gespeicherte Start- und Zielkoordinate des Halbzuges in der üblichen Schachnotation aus (vgl. hierzu die detaillierten Ausführungen auf Seite 2-67!).

In Zeile 6400 überprüfen wir den Inhalt von st, um jeweils zwei Halbzüge durch eine Leerzeile voneinander abzugrenzen (st = 1 nach einem Computer-Halbzug!).

Mit der Anweisung in Zeile 6410 reaktivieren wir den Gesamtbildschirm als Ausgabeeinheit, um anschließend das Vorzeichen der Status-Variablen st umzukehren (Zeile 6420). Außerdem setzen wir in Zeile 6430 den Merker **sm** (swap **m**ate) auf 0, um damit zu gewährleisten, daß der in eine Dame umgewandelte Bauer als endgültig umgewandelt gilt. (Die Variable sm erhält in der Subroutine "Interne Zugausführung" den Wert 1, wenn ein Bauer die Grundlinie des Gegners erreicht und deshalb in eine Dame umgewandelt wurde, um in der Subroutine "Interne Zugzurücknahme" erkennen zu können, daß diese Dame zurückverwandelt werden muß.)

Wenn eine Matt- oder Patt-Situation vorliegt (ma > 0), erhält die Status-Variable st den Wert 0 zugewiesen (Zeile 6440), um nach Rückkehr in die Steuerleiste (Zeile 6450) das Spielende auszulösen.

Programmausschnitt (24):

```
6500 REM *** Schachfeld setzen ***
6510 :
6520 x = INT( pf / 10 )
6530 y = pf MOD 10
6540 PAPER#3, ( ( x + y ) MOD 2 ) * 2
6550 x = 4 * x - 2
6560 y = 25 - 3 * y
6570 WINDOW#3, x, x + 3, y, y + 2
6580 WINDOW SWAP 3,0
6590 CLS
6600 PEN 1
6610 LOCATE 2, 2
6620 PRINT g( f(pf) + 9 )
6630 WINDOW SWAP 3,0
6640 RETURN
6650 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (24):

Die Subroutine "Schachfeld setzen" dient der Ausgabe jeweils eines Feldes **pf** (**p**osition of a **f**ield) des Schachbretts, ggf. mit der zugewiesenen "Figur".

In den Zeilen 6520 und 6530 trennen wir die in der Variablen pf gespeicherte Zahl für die computer-interne Feldbenennung (vgl. Abb. 6 auf Seite 2-15!) in einen Spalten- (x) und in einen Zeilenwert (y). Wir benötigen die getrennten Werte u. a. zur Bestimmung der Farbe des Feldes (Zeile 6540): Wenn x + y (= Quersumme von pf) einen geraden Wert ergibt, so ist dafür ein dunkles Feld, wenn x + y hingegen einen ungeraden Wert ergibt, so ist dafür ein helles Feld auszugeben. Die Formel zur Berechnung der Farbe (Zeile 6540) ist so ausgelegt, daß für ein dunkles Feld die Farbnummer 0 und für ein helles Feld die Farbnummer 2 entsteht.

Dann modifizieren wir in den Zeilen 6550 und 6560 die Werte für x und y, um damit in Zeile 6570 ein WINDOW#3 für ein Schachfeld definieren zu können, welches in Zeile 6580 für Ausgabezwecke kurzfristig aktiviert und gelöscht (Zeile 6590) wird. In dieses nur sehr kurzfristig aktive "Fenster" schreiben wir (Zeilen 6600 - 6620) einen Buchstaben, der die Figur des Feldes repräsentiert (vgl. auch die Ausführungen auf Seite 2-29 und die Abb. 5 auf Seite 2-06!).

Anschließend reaktivieren wir wieder den Gesamtbildschirm als Ausgabeeinheit (Zeile 6630), um mit der Anweisung in Zeile 6640 zu jener Stelle zurückzukehren, von der diese Subroutine aufgerufen wurde.

Programmausschnitt (25):

```
6700 REM *** Interne Zugausfuehrung ***
6710 :
6720 IF pm(3) = 0 GOTO 6760
6730 IF pm(4) = 0 GOTO 6750
6740 f( pm(4) ) = f( pm(3) )
6750 f( pm(3) ) = 0
6760 f0 = f( pm(2) )
6770 f( pm(2) ) = f( pm(1) )
6780 f( pm(1) ) = 0
6790 c = 4.5 - st * 3.5
6800 IF pm(2) MOD 10 <> c GOTO 6840
6810 IF f( pm(2) ) <> 2 * st GOTO 6840
6820 sm = 1
6830 f( pm(2) ) = 7 * st
6840 RETURN
6850 :
6900 REM *** Interne Zugzuruecknahme **
6910 :
6920 IF pm(3) = 0 GOTO 6980
6930 IF pm(4) = 0 GOTO 6970
6940 f( pm(3) ) = 5 * st
6950 f( pm(4) ) = 0
6960 GOTO 6980
6970 f( pm(3) ) = -1 * st
6980 f( pm(1) ) = f( pm(2) )
6990 f( pm(2) ) = f0
7000 IF sm = 0 GOTO 7030
7010 sm = 0
7020 f( pm(1) ) = 2 * st
7030 RETURN
7040 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (25):

Im nebenstehenden Programmausschnitt finden wir die schon an mehreren Stellen erwähnten Subroutinen "Interne Zugausführung" und "Interne Zugzurücknahme", mit deren Hilfe wir jeden berechneten Halbzug, der in pm(1) bis pm(4) gespeichert ist, ausführen (Zeilen 6700 - 6780) und zumeist auch wieder zurücknehmen lassen (Zeilen 6900 - 6990).

Darüber hinaus überprüfen wir in den Zeilen 6790 bis 6810, ob ein Bauer die gegnerische Grundlinie erreicht hat, um ihn dort in eine Dame umzuwandeln (Zeile 6830). Außerdem wird ein Schalter **sm** (**s**wap **m**ate) auf 1 gesetzt, damit bei der Zurücknahme des Zuges die Dame wieder in einen Bauern zurückverwandelt werden kann (Zeilen 7000 - 7020).

Programmausschnitt (26):

```
7100 REM *** Rochadenueberpruefung ***
7110 :
7120 fu = 1
7130 fl = 0
7140 ro = SGN( pm(2) - pm(1) ) * 10
7150 pm(3) = 49.5 + 3.5 * ( st + ro )
7160 pm(4) = pm(1) + ro
7170 IF f( pm(3) ) <> 5 * st GOTO 7310
7180 FOR aa=pm(4) TO pm(3)-ro STEP ro
7190 IF f(aa) <> 0 THEN fl = 1
7200 NEXT aa
7210 IF fl = 1 GOTO 7320
7220 FOR ba=pm(1) TO pm(3) STEP ro
7230 FOR m=3 TO 9
7240 IF fl = 0 THEN pm=ba : GOSUB 7600
7250 NEXT m
7260 FOR aa=1 TO 2
7270 pm = st * ( jp(aa) - (aa-1) * 22 )
7280 IF f(pm + ba) = -2 * st THEN fl=1
7290 NEXT aa, ba
7300 GOTO 7320
7310 fl = 1
7320 RETURN
7330 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (26):

Zu Beginn des Funktionsmoduls **Computerzug** (vgl. Programmausschnitt (13), Seite 2-47!) untersuchten wir in der Zeile 4370, ob der König noch rochadefähig ist. Im Bejahungsfall wurde aus einer FOR ... NEXT-Schleife, mit der wir den jeweiligen Königszug der Rochade erzeugten, die hier zu erläuternde Subroutine aufgerufen. In ihr finden verschiedene Überprüfungen statt.

Zunächst setzen wir fu auf 1 (Zeile 7120), damit in der weiter unten aufgerufenen Routine Feldbestimmung zu einer Stelle verzweigt wird, an der wir kontrollieren, ob ein durch die Rochade angesprochenes Feld bedroht ist, weil sie dann gem. den Schachregeln nicht ausgeführt werden dürfte. Der Merker fl erhält den Wert 0 (Zeile 7130), der nur dann auf 1 gesetzt wird, wenn die Ausführung der Rochade unzulässig ist.

Nachdem in Zeile 7140 die Schrittweite der an der Rochade beteiligten Figuren ermittelt worden ist (große Rochade: ro = -10, kleine Rochade: ro = 10), legen wir in pm(3) und pm(4) die Start- und Zielkoordinate des Turmzugs ab und überprüfen in Zeile 7170, ob der von der Rochade angesprochene Turm noch rochadefähig ist.

In der sich anschließenden FOR ... NEXT-Schleife (Zeilen 7180 - 7200) kontrollieren wir, ob die zwischen König und Turm liegenden Felder unbesetzt sind: Ist dies nicht der Fall, wird in Zeile 7210 nach 7310 verzweigt, wo der Merker fl den Wert 1 erhält.

Wir erwähnten bereits, daß eine Rochade nur ausgeführt werden darf, wenn die durch sie angesprochenen Felder nicht bedroht sind. Die entsprechende Untersuchung führen wir in den Zeilen 7220 bis 7290 folgendermaßen durch: Ausgehend von dem jeweils zu untersuchenden Feld ba simulieren wir die möglichen Züge sämtlicher Figuren (mit Ausnahme des Bauern) durch Aufruf der Subroutine Feldbestimmung in Zeile 7240. Eine Bedrohungssituation liegt dann vor (fl = 1), wenn die "Simulationsfigur" auf eine gleichartige Figur des Gegners stößt. Die Überprüfung der evtl. Bedrohung durch einen Bauern erfolgt in den Zeilen 7260 - 7290.

Programmausschnitt (27):

```
7400 REM *** Koenig im Schach ? ***
7410 :
7420 fu = 1
7430 fl = 0
7440 FOR aa=11 TO 88
7450 FOR ba=8 TO 9
7460 IF f(aa) = st * ba THEN pm = aa
7470 NEXT ba, aa
7480 FOR m=3 TO 9
7490 IF fl = 0 THEN GOSUB 7600
7500 NEXT m
7510 FOR aa=1 TO 2
7520 c = st * ( jp(aa) - (aa-1) * 22 )
7530 IF f(pm + c) = -1 * st THEN fl=1
7540 IF f(pm + c) = -2 * st THEN fl=1
7550 NEXT aa
7560 RETURN
7570 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (27):

In der Subroutine "König im Schach ?" setzen wir zunächst fu auf 1 (Zeile 7420), damit in der weiter unten aufgerufenen Routine Feldbestimmung zu einer Stelle verzweigt wird, in der wir kontrollieren, ob das Feld des Königs bedroht ist. Der Merker fl erhält den Wert 0 (Zeile 7430), der nur dann auf 1 gesetzt wird, wenn der König bedroht wird.

Nachdem wir in der folgenden FOR ... NEXT-Schleife (Zeilen 7440 - 7470), in die eine weitere FOR ... NEXT-Schleife (Zeile 7450) eingebettet ist, die Position des Königs (pm) ermittelt haben, simulieren wir für dieses Feld die möglichen Züge sämtlicher Figuren mit Ausnahme des Bauern (Zeilen 7480 - 7500) durch Aufruf der Subroutine Feldbestimmung in der Zeile 7490. Eine Bedrohungssituation liegt dann vor (fl = 1), wenn die "Simulationsfigur" auf eine gleichartige Figur des Gegners stößt. Die Überprüfung der evtl. Bedrohung durch einen Bauern erfolgt in den Zeilen 7510 bis 7550.

Programmausschnitt (28):

```
7600 REM *** Feldbestimmung ***
7610 :
7620 FOR bb=-1 TO 1 STEP 2
7630 FOR ab=jm(m,1) TO jm(m,2)
7640 c = pm + jp(ab) * nj(m) * bb
7650 pf = pm
7660 pf = pf + jp(ab) * bb
7670 IF pf < 11 OR pf > 88 GOTO 7730
7680 IF SGN(f(pf))=SGN(f(pm)) GOTO 7730
7690 IF f(pf) = 10 GOTO 7730
7700 ON fu GOSUB 7800, 7900, 8200, 8500
7710 IF pf = c GOTO 7730
7720 IF f(pf) = 0 GOTO 7660
7730 NEXT ab
7740 NEXT bb
7750 RETURN
7760 :
7800 REM *** Feldueberpruefung ***
7810 :
7820 IF f(pf) = -m * st THEN f1 = 1
7830 RETURN
7840 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (28):

Die Subroutine Feldbestimmung dient dazu, für jede beliebige Figur (m) alle von einer bestimmten Position (pm) aus erreichbaren Felder (pf) zu ermitteln. Um die festgelegten Schrittweiten bei den Berechnungen sowohl zu subtrahieren als auch zu addieren (vgl. hierzu auch unser Beispiel auf Seite 2-23!), haben wir die FOR ... NEXT-Schleife der Zeilen 7630 bis 7730 durch die FOR ... NEXT-Schleife der Zeilen 7620 bis 7740 eingerahmt . Abhängig vom Inhalt der Variablen fu, der in einem übergeordneten Modul festgelegt wurde, rufen wir in Zeile 7700 unterschiedliche Subroutinen auf, um dort die Bearbeitung für das ermittelte Feld fortzusetzen.

Die Routine Feldüberprüfung, die aus Zeile 7700 der "Feldbestimmung" aufgerufen wird, wenn fu den Wert 1 hat, dient der Überprüfung, ob ein von einer Figur (m) erreichbares Feld mit der Nummer pf mit einer gleichartigen Figur des Gegners belegt ist.

Beispiele:

(1) **st** = **1** ("Computer am Zug")
m = **3** (Springer des Computers, weil **st** = **1**)
pf = **36** (Nr. eines vom Springer des Computers erreichbaren Feldes, das in "Feldbestimmung" ermittelt wurde)
Die Untersuchung, ob **f(36)** einen Springer des Spielers (**-3**) enthält, und zwar durch den Befehl
IF f(36) = -3 * 1 ... ,
führt dazu, daß dem Merker **fl** der Wert **1** zugewiesen wird, wenn **f(36)** = **-3** (Springer des Spielers).

(2) **st** = **-1** ("Spieler am Zug")
m = **3** (Springer des Spielers, weil **st** = **-1**)
pf = **54** (Nr. eines vom Springer des Spielers erreichbaren Feldes, das in "Feldbestimmung" ermittelt wurde)
Die Untersuchung, ob **f(54)** einen Springer des Computers (**3**) enthält, und zwar durch den Befehl
IF f(54) = -3 * -1 ... ,
führt dazu, daß dem Merker **fl** der Wert **1** zugewiesen wird, wenn **f(54)** = **3** (Springer des Computers).

Programmausschnitt (29):

```
7900 REM *** Zugerzeuger Computerzug **
7910 :
7920 IF m > 2 GOTO 8080
7930 IF pm - pf <> 1 GOTO 8020
7940 IF f(pf) <> 0 GOTO 8110
7950 IF m = 2 GOTO 8080
7960 IF f( pf - 1 ) <> 0 GOTO 8080
7970 fm = fm + 1
7980 fm(fm, 1) = pm
7990 fm(fm, 2) = pf - 1
8000 GOTO 8080
8010 :
8020 IF pm - pf = 11 GOTO 8040
8030 IF pf - pm <> 9 GOTO 8110
8040 IF pm MOD 10 <> 4 GOTO 8070
8050 IF f( pf + 1 ) <> -1 GOTO 8070
8060 IF f(pf) = 0 THEN 8080 ELSE 8110
8070 IF f(pf) >= 0 GOTO 8110
8080 fm = fm + 1
8090 fm(fm, 1) = pm
8100 fm(fm, 2) = pf
8110 RETURN
8120 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (29):

Wenn der Rechner im Funktionsmodul **Computerzug** eine ihm gehörende Figur gefunden hat (vgl. den Programmausschnitt (13) auf Seite 2-46!), wird dort in Zeile 4150 die Subroutine **Feldbestimmung** aufgerufen, in der wiederum die hier zu erläuternden Befehle herangezogen werden, sofern **fu = 2.**

Da in der Routine Feldbestimmung die Halbzüge eines Bauern noch nicht fehlerfrei bestimmt worden sind, müssen wir für sie in diesem Abschnitt noch besondere Untersuchungen durchführen. Für alle anderen Figuren verzweigen wir aufgrund der Anweisung in Zeile 7920 direkt zum Befehl der Zeile 8080 und erhöhen dort den Zugzähler fm um 1 (vgl. hierzu Seite 2-47!), um dann die Startkoordinate in fm(.., 1) und die Zielkoordinate in fm(.., 2) abzulegen (Zeilen 8090 - 8100).

In den Zeilen 7930 bis 7960 untersuchen wir, ob der Bauer neben dem Einzelschritt auch einen Doppelschritt durchführen kann, um hierfür im Bejahungsfall die Koordinaten mit den Befehlen der Zeilen 7970 bis 7990 zu speichern.

Da für den Bauern möglicherweise auch unzulässige Schritte (waagerechte oder zurückschreitende) erzeugt worden sind, müssen wir diese mit den Abfragen der Zeilen 8020 und 8030 ggf. unterdrücken. Gleichzeitig ermitteln wir damit die schräge verlaufenden Züge, für die dann untersucht wird, ob ein "en-passant-Schlag" (Zeilen 8040 - 8060) oder ein normaler Schlagzug (Zeile 8070) vorliegt. Ist dies der Fall, erfolgt dessen Speicherung wie für jede andere Figur in den Zeilen 8080 bis 8100.

Programmausschnitt (30):

```
8200 REM *** Zugerzeuger Spielerzug ***
8210 :
8220 IF m > 2 GOTO 8380
8230 IF pf - pm <> 1 GOTO 8320
8240 IF f(pf) <> 0 GOTO 8400
8250 IF m = 2 GOTO 8380
8260 IF f( pf + 1 ) <> 0 GOTO 8380
8270 IF pm MOD 10 <> 2 GOTO 8380
8280 mm = mm + 1
8290 mm(mm) = pf + 1
8300 GOTO 8380
8310 :
8320 IF pf - pm = 11 GOTO 8340
8330 IF pm - pf <> 9 GOTO 8400
8340 IF pm MOD 10 <> 5 GOTO 8370
8350 IF f( pf - 1 ) <> 1 GOTO 8370
8360 IF f(pf) = 0 THEN 8380 ELSE 8400
8370 IF -f(pf) >= 0 GOTO 8400
8380 mm = mm + 1
8390 mm(mm) = pf
8400 RETURN
8410 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (30):

Die nebenstehende Subroutine "Zugerzeuger Spielerzug" entspricht weitgehend der in Programmausschnitt (29) vorgestellten Subroutine "Zugerzeuger Computerzug".

Der Unterschied besteht in erster Linie darin, daß diese Befehle nur für eine Figur aufgerufen werden, für die die Startkoordinate "bekannt" ist, so daß lediglich die Zielkoordinate zu speichern ist (vgl. vor allem die Zeilen 8280 - 8290 und 8380 - 8390!).

Darüber hinaus mußte die Anweisung der Zeile 8270 ergänzt werden, mit deren Hilfe wir sicherstellen, daß ein berechneter Doppelzug nur von der Grundposition des Bauern ausgeführt wird. (Da die den Bauern repräsentierende Nummer nach einem Doppelzug erst im folgenden Halbzug geändert wird, um dadurch sicherzustellen, daß die Möglichkeit eines "en-passant-Schlags" erkannt werden kann (vgl. hierzu Seite 2-77!), wäre es ohne diesen Befehl möglich, daß der Computer beim fingierten Bauernzug des Spielers einen unzulässigen zweiten Doppelschritt des Spielers in "seine Überlegungen" einbezieht.)

Programmausschnitt (31):

```
8500 REM *** Positionswertung ***
8510 :
8520 IF m > 2 GOTO 8670
8530 IF pm - pf <> sg GOTO 8600
8540 IF f(pf) <> 0 GOTO 8680
8550 IF m = 2 GOTO 8670
8560 IF f( pf - sg ) <> 0 GOTO 8670
8570 v = v + sg * vm(0)
8580 GOTO 8670
8590 :
8600 IF pm - pf = 11 * sg GOTO 8630
8610 IF pf - pm <> 9 * sg GOTO 8680
8620 c0 = 4.5 - 0.5 * sg
8630 IF pm MOD 10 <> c0 GOTO 8660
8640 IF f( pf + sg ) <> -sg GOTO 8660
8650 IF f(pf) = 0 THEN 8670 ELSE 8680
8660 IF f(pf) * sg >= 0 GOTO 8680
8670 v = v + sg * vm( ABS( f(pf) ) )
8680 RETURN
8690 :
```

Erläuterungen zum nebenstehenden Programmausschnitt (31):

Um die Programmlogik zu verdeutlichen, zitieren wir hier aus unseren Erläuterungen zum Programmausschnitt (21) auf Seite 2-75:

"Die ... Schleife (Zeilen 5890 - 5950) **ergänzt** das oben erläuterte Bewertungsverfahren um einen spielstrategischen Aspekt, und zwar ermitteln wir durch Aufruf der Subroutine Feldbestimmung (in Zeile 5940) für **jede** Figur alle erreichbaren Felder und vergeben hier für ein "beherrschtes" leeres Feld einen Punkt und für eine schlagbare Figur ihren Figurenwert . ... Dabei ergeben die Berechnungen für die Figuren des Computers positive und für jene des Spielers negative Zahlen, die zur Bewertungsvariablen v addiert werden."

Wenn fu = 4 ist, verzweigen wir in Zeile 7700 der oben erwähnten Subroutine Feldbestimmung zu den nebenstehenden Befehlen der "Positionswertung". Sie haben wiederum weitgehende Ähnlichkeit mit jenen in der Subroutine "Zugerzeuger Computerzug" und ergänzen die Bewertung der Computer-Halbzüge und die Bewertung der vom Computer fingierten Spieler-Halbzüge. Um eine Bearbeitung sowohl für die Figuren des Computers als auch für jene des Spielers zu ermöglichen, verwenden wir die Variable sg, die nur den Wert 1 oder -1 enthalten kann (vgl. Zeile 5920 im Ausschnitt (21) !).

Die Bewertungsformel der Zeile 8570 bzw. 8670 im einzelnen:

```
v  =  v  +  sg  *  vm( abs( f(pf) ) )
     (1)   (2)           (3)
```

(1) ---> vorläufig ermittelter Stellungswert (vgl. Seite 2-73!),
(2) ---> Vorzeichen der behandelten Figur,
(3) ---> vm(..) enthält den Figurenwert, wobei sich der Index aus dem Absolutwert der Feldbelegung (vgl. Seite 2-21!) ergibt.

Programmausschnitt (32):

```
8990 REM ******************************
9000 REM *          Spielende         *
9010 REM ******************************
9020 :
9030 WINDOW SWAP 1,0
9040 CLS
9050 PRINT
9060 IF ma = 1 THEN PRINT "PATT!"
9070 IF ma = 2 THEN PRINT "MATT!"
9080 PRINT
9090 RETURN
```

Erläuterungen zum nebenstehenden Programmausschnitt (32):

Zu Beginn des Funktionsmoduls **Spielende** aktivieren wir den Bildschirmbereich WINDOW#1 für alle weiteren Ausgaben und löschen den Inhalt des definierten "Fensters".

Wenn bei Spielstärke 1 eine Patt- oder Mattsituation für den Computer eintritt, wird nach einer Leerzeile in Abhängigkeit vom Inhalt der Variablen ma der Text

PATT! oder

MATT! ausgegeben,

während eine Patt- oder Mattsituation des Spielers daran zu erkennen ist, daß der Rechner keine Zugeingabe mehr akzeptiert.

Bei Spielstärke 2 hingegen erkennt das System die Patt- bzw. Mattsituation sowohl für den Rechner als auch für den Spieler und gibt dann in jedem Fall eine der obigen Nachrichten aus.

3. Vollständige Liste des BASIC-Schachprogramms

```
10 REM ********************************
20 REM *                              *
30 REM *   Schachprogramm in BASIC    *
40 REM *                              *
50 REM ********************************
60 REM
70 REM ********************************
80 REM *         Steuerleiste         *
90 REM ********************************
100 :
200 GOSUB 1000  : REM Initialisierung
210 :
300 WHILE st=-1
310 GOSUB 2000  : REM Spielereingabe
320 GOSUB 6000  : REM Zugausgabe
330 WEND
340 :
400 WHILE st=1
410 GOSUB 4000  : REM Computerzug
420 GOSUB 6000  : REM Zugausgabe
430 WEND
440 :
500 IF st <> 0 GOTO 300
510 :
600 GOSUB 9000  : REM Spielende
610 :
700 END
```

```
990  REM ******************************
1000 REM *      Initialisierung       *
1010 REM ******************************
1020 :
1030 DEFINT a, b, c, f, j, m, n, p, s
1040 DEFINT v, x, y
1050 DEFSTR i, g
1060 :
1070 DIM vm(10), f(99), jp(8), jm(9,2)
1080 DIM nj(9), pm(8), g(18), fm(120,3)
1090 DIM mm(27)
1100 :
1110 FOR a=0 TO 9 : READ vm(a) : NEXT a
1120 DATA 1, 5, 5, 15, 15, 25, 25
1130 DATA 50, 2000,2000
1140 :
1150 FOR a=0 TO 99 : READ f(a) : NEXT a
1160 DATA 10,10,10,10,10,10,10,10,10,10
1170 DATA 10,-5,-1, 0, 0, 0, 0, 1, 5,10
1180 DATA 10,-3,-1, 0, 0, 0, 0, 1, 3,10
1190 DATA 10,-4,-1, 0, 0, 0, 0, 1, 4,10
1200 DATA 10,-7,-1, 0, 0, 0, 0, 1, 7,10
1210 DATA 10,-8,-1, 0, 0, 0, 0, 1, 8,10
1220 DATA 10,-4,-1, 0, 0, 0, 0, 1, 4,10
1230 DATA 10,-3,-1, 0, 0, 0, 0, 1, 3,10
1240 DATA 10,-5,-1, 0, 0, 0, 0, 1, 5,10
1250 DATA 10,10,10,10,10,10,10,10,10,10
1260 :
1270 FOR a=1 TO 8 : READ jp(a) : NEXT a
1280 DATA 9, 11, 10, 1, 12, 21, 8, 19
1290 :
1300 FOR a=1 TO 9 : READ nj(a) : NEXT a
1310 DATA 1, 1, 1, 7, 7, 7, 7, 1, 1
1320 :
```

```
1330 FOR a=1 TO 9 : FOR b=1 TO 2
1340 READ jm(a,b)
1350 NEXT b, a
1360 DATA 1,4, 1,4, 5,8, 1,2, 3,4
1370 DATA 3,4, 1,4, 1,4, 1,4
1380 :
1390 FOR a=0 TO 18 : READ g(a) : NEXT a
1400 DATA K, K, D, T, T, L, S, B, B, ""
1410 DATA b, b, s, l, t, t, d, k, k
1420 :
1430 MODE 1
1440 INK 1, 24
1450 INK 2, 8
1460 :
1470 PRINT TAB(9) "Schneider-";
1480 PRINT "Schachprogramm"
1490 LOCATE 16, 3
1500 PRINT "in BASIC"
1510 LOCATE 12, 5
1520 PRINT "******************"
1530 LOCATE 3, 11
1540 PRINT "Bitte waehlen Sie ..."
1550 LOCATE 3, 15
1560 PRINT "Spielstaerke 1"
1570 LOCATE 3, 16
1580 PRINT "(ca.  3 Minuten ";
1590 PRINT "bis zum Zug)  ... (1)"
1600 LOCATE 3, 18
1610 PRINT "Spielstaerke 2"
1620 LOCATE 3, 19
1630 PRINT "(ca. 10 Minuten ";
1640 PRINT "bis zum Zug)  ... (2)"
1650 LOCATE 14, 23
1660 PRINT "... ´1´ oder ´2´ druecken!"
1670 in = INKEY$
1680 IF in < "1" OR in > "2" GOTO 1670
1690 mo = VAL(in)
1700 :
```

```
1710 CLS
1720 FOR a=1 TO 8
1730 PRINT
1740 PRINT CHR$( 57 - a )
1750 PRINT
1760 NEXT a
1770 TAG
1780 MOVE 9, 12
1790 FOR a=1 TO 8
1800 PRINT "  ";
1810 PRINT CHR$( 64 + a ); " ";
1820 NEXT a
1830 TAGOFF
1840 FOR a=10 TO 80 STEP 10
1850 FOR b=1 TO 8
1860 pf = a + b
1870 GOSUB 6500
1880 NEXT b, a
1890 :
1900 WINDOW#1, 35, 40, 21, 25
1910 WINDOW#2, 35, 40, 1, 19
1920 LOCATE#2, 1,19
1930 bm = 11
1940 st = -1
1950 ro = 0
1960 ma = 0
1970 RETURN
1980 :
1990 REM ******************************
2000 REM *      Spielereingabe        *
2010 REM ******************************
2020 :
2030 WINDOW SWAP 1,0
2040 fl = 0
2050 ro = 0
2060 LOCATE 1,1
2070 PRINT "Ihr"
```

```
2080 PRINT " Zug: "
2090 PRINT CHR$(143); "_-__ "
2100 LOCATE 1,4
2110 a = 1
2120 in = INKEY$
2130 IF in = CHR$(127) GOTO 2260
2140 IF in < "a" OR in > "h" GOTO 2120
2150 PRINT UPPER$(in);
2160 PRINT CHR$(143); CHR$(8);
2170 pm(a) = ( ASC(in)-96 ) * 10
2180 in = INKEY$
2190 IF in = CHR$(127) GOTO 2260
2200 IF in < "1" OR in > "8" GOTO 2180
2210 PRINT UPPER$(in);
2220 IF a = 1 THEN PRINT "-";
2230 PRINT CHR$(143); CHR$(8);
2240 pm(a) = pm(a) + VAL(in)
2250 IF a = 1 THEN a = 2 : GOTO 2120
2260 IF in = CHR$(127) GOTO 2060
2270 pm(3) = 0
2280 pm(4) = 0
2290 in = INKEY$
2300 IF in = CHR$(127) GOTO 2040
2310 IF in <> CHR$(13) GOTO 2290
2320 PRINT " "
2330 IF f( pm(1) ) >= 0 GOTO 2040
2340 IF pm(1) <> 51 GOTO 2380
2350 IF f(51) <> -8 GOTO 2380
2360 IF ABS(pm(1)-pm(2)) <>20 GOTO 2380
2370 ro = 1
2380 m = -f( pm(1) )
2390 :
2400 ON ro + 1 GOSUB 3000, 7100
2410 IF fl = 1 GOTO 2040
2420 IF ro <> 0 THEN 2470
2430 GOSUB 6700
2440 GOSUB 7400
```

```
2450 GOSUB 6900
2460 IF fl = 1 GOTO 2040
2470 WINDOW SWAP 1,0
2480 RETURN
2490 :
3000 REM *** Zugueberpruefung ***
3010 :
3020 IF pm(2) = pm(1) GOTO 3190
3030 jp = 0
3040 FOR a=jm(m,1) TO jm(m,2)
3050 IF jp <> 0 GOTO 3090
3060 jp = jp(a)
3070 c = ( pm(2)-pm(1) ) MOD jp
3080 IF c <> 0 THEN jp = 0
3090 NEXT a
3100 IF jp = 0 GOTO 3190
3110 jp = jp * SGN( pm(2)-pm(1) )
3120 FOR a=pm(1)+jp TO pm(2)-jp STEP jp
3130 IF f(a) <> 0 THEN fl = 1
3140 NEXT a
3150 IF fl = 1 GOTO 3200
3160 IF f( pm(2) ) < 0 GOTO 3190
3170 IF m < 4 OR m > 7 THEN GOSUB 3300
3180 GOTO 3200
3190 fl = 1
3200 RETURN
3210 :
3300 REM *** Sonderueberpruefungen ***
3310 :
3320 ON m GOSUB 3400, 3500
3330 IF m > 2 THEN GOSUB 3700
3340 RETURN
3350 :
3400 REM *** Bauer (1) ***
3410 :
3420 IF pm(2) - pm(1) <> 2 GOTO 3450
3430 IF f( pm(2) ) <> 0 THEN fl = 1
```

```
3440 GOTO 3460
3450 GOSUB 3500
3460 RETURN
3470 :
3500 REM *** Bauer (2) ***
3510 :
3520 IF ABS( pm(2)-pm(1) )=10 GOTO 3600
3530 IF pm(2) - pm(1) <> 1 GOTO 3560
3540 IF f( pm(2) ) = 0 GOTO 3610
3550 GOTO 3600
3560 IF f( pm(2) ) > 0 GOTO 3610
3570 IF pm(1) MOD 10 <> 5 GOTO 3600
3580 pm(3) = pm(2) - 1
3590 IF f( pm(3) ) = 1 GOTO 3610
3600 fl = 1
3610 GOSUB 3700
3620 RETURN
3630 :
3700 REM *** nur ein Schritt ! ***
3710 :
3720 IF pm(2) - pm(1) <> jp THEN fl = 1
3730 RETURN
3740 :
3990 REM ******************************
4000 REM *          Computerzug        *
4010 REM ******************************
4020 :
4030 fu = 2
4040 ro = 0
4050 fm = 0
4060 WINDOW SWAP 1,0
4070 LOCATE 1,1
4080 PRINT "Ich"
4090 PRINT "werte:"
4100 PRINT "__-__"
4110 FOR pm=11 TO 88
4120 IF f(pm) <= 0 GOTO 4160
```

```
4130 IF f(pm) = 10 GOTO 4160
4140 m = f(pm)
4150 GOSUB 7600
4160 NEXT pm
4170 IF f(58) <> 8 GOTO 4270
4180 pm(1) = 58
4190 FOR a=38 TO 78 STEP 40
4200 pm(2) = a
4210 GOSUB 7100
4220 IF fl = 1 GOTO 4260
4230 fm = fm + 1
4240 fm(fm, 1) = pm(1)
4250 fm(fm, 2) = pm(2)
4260 NEXT a
4270 FOR a=1 TO fm
4280 pm(1) = fm(a, 1)
4290 pm(2) = fm(a, 2)
4300 GOSUB 5100
4310 GOSUB 5400
4320 GOSUB 6700
4330 GOSUB 7400
4340 v = -9990
4350 IF fl = 0 THEN GOSUB 5800
4360 fm(a, 3) = v
4370 GOSUB 6900
4380 NEXT a
4390 GOSUB 5600
4400 IF fm(1,3) > -9990 GOTO 4440
4410 GOSUB 7400
4420 IF fl = 1 THEN ma = 2 ELSE ma = 1
4430 GOTO 5060
4440 FOR a=fm TO 1 STEP -1
4450 IF fm(a,3) = -9990 THEN fm = a - 1
4460 NEXT a
4470 IF mo = 1 GOTO 5010
4480 v0 = -9999
4490 FOR a=1 TO fm
```

```
4500 fm(a, 3) = 9999
4510 pm(1) = fm(a, 1)
4520 pm(2) = fm(a, 2)
4530 st = 1
4540 GOSUB 5100
4550 GOSUB 5400
4560 GOSUB 6700
4570 FOR b=1 TO 4
4580 pm( b + 4 ) = pm(b)
4590 NEXT b
4600 fh = f0 : sh = sm : sm = 0
4610 pc = bm
4620 st = -1
4630 WHILE pc <= bm + 99
4640 mm = 0
4650 pm = pc MOD 100
4660 IF f(pm) >= 0 GOTO 4910
4670 m = -f(pm)
4680 fu = 3
4690 GOSUB 7600
4700 b = 1
4710 WHILE b <= mm
4720 pm(1) = pc MOD 100
4730 pm(2) = mm(b)
4740 GOSUB 5100
4750 GOSUB 6700
4760 GOSUB 7400
4770 IF f1 = 1 GOTO 4850
4780 GOSUB 5800
4790 IF v > v0 GOTO 4820
4800 bm = pc MOD 100
4810 b = mm : pc = 200
4820 IF fm(a, 3) = -9999 GOTO 4840
4830 IF v >= fm(a, 3) GOTO 4850
4840 fm(a, 3) = v
4850 GOSUB 6900
4860 IF f1 = 0 GOTO 4890
```

```
4870 GOSUB 7400
4880 IF f1 = 0 THEN fm(a, 3) = -9999
4890 b = b + 1
4900 WEND
4910 IF pc=88 THEN pc=111 ELSE pc=pc+1
4920 WEND
4930 IF v0 < fm(a,3) THEN v0 = fm(a,3)
4940 FOR b=1 TO 4
4950 pm(b) = pm( b + 4 )
4960 NEXT b
4970 f0 = fh : sm = sh : st = 1
4980 GOSUB 6900
4990 NEXT a
5000 GOSUB 5600
5010 pm(1) = fm(1, 1)
5020 pm(2) = fm(1, 2)
5030 GOSUB 5100
5040 IF fm(1, 3) =  9999 THEN ma = 2
5050 IF fm(1, 3) = -9999 THEN ma = 1
5060 WINDOW SWAP 1,0
5070 st = 1
5080 RETURN
5090 :
5100 REM *** Zugparameter einstellen **
5110 :
5120 IF f(pm(2))=0 THEN f1=0 ELSE f1=1
5130 pm(3) = 0
5140 pm(4) = 0
5150 IF f( pm(1) ) <> 2 * st GOTO 5240
5160 c = 4.5 - st * 0.5
5170 IF pm(1) MOD 10 <> c GOTO 5240
5180 IF pm(1)-pm(2) = 11*st GOTO 5200
5190 IF pm(2)-pm(1) <> 9*st GOTO 5290
5200 pm(3) = pm(2) + st
5210 f1 = 1
5220 GOTO 5290
5230 :
```

```
5240 IF f( pm(1) ) <> 8 * st GOTO 5290
5250 ro = ( pm(2) - pm(1) ) / 2
5260 IF ABS(ro) <> 10 GOTO 5290
5270 pm(3) = 49.5 + 3.5 * ( st + ro )
5280 pm(4) = pm(1) + ro
5290 RETURN
5300 :
5400 REM *** Zugausgabe in WINDOW#1 ***
5410 :
5420 LOCATE 1,4
5430 FOR aa = 1 TO 2
5440 PRINT CHR$( INT( pm(aa)/10 ) +64);
5450 PRINT CHR$( pm(aa) MOD 10 + 48 );
5460 IF aa = 2 GOTO 5490
5470 IF fl = 0 THEN PRINT "-";
5480 IF fl = 1 THEN PRINT "x";
5490 NEXT aa
5500 RETURN
5510 :
5600 REM *** Sortieren von fm(a,b) ***
5610 :
5620 LOCATE 1,4
5630 PRINT "__-__"
5640 FOR a=1 TO fm - 1
5650 FOR b=a + 1 TO fm
5660 IF fm(a, 3) >= fm(b, 3) GOTO 5720
5670 FOR c=1 TO 3
5680 f0 = fm(a, c)
5690 fm(a, c) = fm(b, c)
5700 fm(b, c) = f0
5710 NEXT c
5720 NEXT b, a
5730 RETURN
5740 :
5800 REM *** Bewertungsfunktion ***
5810 :
5820 fu = 4
```

```
5830 v = 0
5840 sg = SGN(f0)
5850 v = v - 2 * vm( f0 * sg ) * sg
5860 sg = SGN(fh)
5870 v = v - 2 * vm( fh * sg ) * sg
5880 :
5890 FOR pm=11 TO 88
5900 IF f(pm) = 0 GOTO 5950
5910 IF f(pm) = 10 GOTO 5950
5920 sg = SGN( f(pm) )
5930 m = ABS( f(pm) )
5940 GOSUB 7600
5950 NEXT pm
5960 RETURN
5970 :
5990 REM *****************************
6000 REM *        Zugausgabe         *
6010 REM *****************************
6020 :
6030 GOSUB 6700
6040 IF fm(1, 3) = -9990 GOTO 6430
6050 IF pm(4) = 0 THEN c = 3 ELSE c = 4
6060 IF pm(3) = 0 THEN c = 2
6070 FOR a=1 TO c
6080 pf = pm(a)
6090 GOSUB 6500
6100 NEXT a
6110 FOR a = 5 TO 8 STEP 3
6120 IF f( pm(2) ) <> a * st GOTO 6140
6130 f( pm(2) ) = a * st + st
6140 NEXT a
6150 c = 4.5 - st * 0.5
6160 FOR a=10 + c TO 80 + c STEP 10
6170 IF f(a) <> -st GOTO 6190
6180 f(a) = f(a) - st
6190 NEXT a
6200 c = c + st * 2
```

```
6210 IF f( pm(2) ) <> st GOTO 6250
6220 IF pm(2) MOD 10 <> c GOTO 6250
6230 f( pm(2) ) = f( pm(2) ) + st
6240 :
6250 WINDOW SWAP 2,0
6260 IF pm(4) <> 0 GOTO 6380
6270 f1 = 0
6280 IF f0 <> 0 OR pm(3) <> 0 THEN f1=1
6290 FOR a=1 TO 2
6300 PRINT CHR$( INT( pm(a)/10 ) + 64);
6310 PRINT CHR$( pm(a) MOD 10 + 48 );
6320 IF a = 2 GOTO 6350
6330 IF f1 =  0 THEN PRINT "-";
6340 IF f1 <> 0 THEN PRINT "x";
6350 NEXT a
6360 PRINT
6370 GOTO 6400
6380 IF ro = -10 THEN PRINT "0-0-0"
6390 IF ro =  10 THEN PRINT " 0-0 "
6400 IF st = 1 THEN PRINT
6410 WINDOW SWAP 2,0
6420 st = -st
6430 sm = 0
6440 IF ma > 0 THEN st = 0
6450 RETURN
6460 :
6500 REM *** Schachfeld setzen ***
6510 :
6520 x = INT( pf / 10 )
6530 y = pf MOD 10
6540 PAPER#3, ( ( x + y ) MOD 2 ) * 2
6550 x = 4 * x - 2
6560 y = 25 - 3 * y
6570 WINDOW#3, x, x + 3, y, y + 2
6580 WINDOW SWAP 3,0
6590 CLS
6600 PEN 1
6610 LOCATE 2, 2
```

```
6620 PRINT g( f(pf) + 9 )
6630 WINDOW SWAP 3,0
6640 RETURN
6650 :
6700 REM *** Interne Zugausfuehrung ***
6710 :
6720 IF pm(3) = 0 GOTO 6760
6730 IF pm(4) = 0 GOTO 6750
6740 f( pm(4) ) = f( pm(3) )
6750 f( pm(3) ) = 0
6760 f0 = f( pm(2) )
6770 f( pm(2) ) = f( pm(1) )
6780 f( pm(1) ) = 0
6790 c = 4.5 - st * 3.5
6800 IF pm(2) MOD 10 <> c GOTO 6840
6810 IF f( pm(2) ) <> 2 * st GOTO 6840
6820 sm = 1
6830 f( pm(2) ) = 7 * st
6840 RETURN
6850 :
6900 REM *** Interne Zugzuruecknahme **
6910 :
6920 IF pm(3) = 0 GOTO 6980
6930 IF pm(4) = 0 GOTO 6970
6940 f( pm(3) ) = 5 * st
6950 f( pm(4) ) = 0
6960 GOTO 6980
6970 f( pm(3) ) = -1 * st
6980 f( pm(1) ) = f( pm(2) )
6990 f( pm(2) ) = f0
7000 IF sm = 0 GOTO 7030
7010 sm = 0
7020 f( pm(1) ) = 2 * st
7030 RETURN
7040 :
7100 REM *** Rochadenueberpruefung ***
7110 :
```

```
7120 fu = 1
7130 fl = 0
7140 ro = SGN( pm(2) - pm(1) ) * 10
7150 pm(3) = 49.5 + 3.5 * ( st + ro )
7160 pm(4) = pm(1) + ro
7170 IF f( pm(3) ) <> 5 * st GOTO 7310
7180 FOR aa=pm(4) TO pm(3)-ro STEP ro
7190 IF f(aa) <> 0 THEN fl = 1
7200 NEXT aa
7210 IF fl = 1 GOTO 7320
7220 FOR ba=pm(1) TO pm(3) STEP ro
7230 FOR m=3 TO 9
7240 IF fl = 0 THEN pm=ba : GOSUB 7600
7250 NEXT m
7260 FOR aa=1 TO 2
7270 pm = st * ( jp(aa) - (aa-1) * 22 )
7280 IF f(pm + ba) = -2 * st THEN fl=1
7290 NEXT aa, ba
7300 GOTO 7320
7310 fl = 1
7320 RETURN
7330 :
7400 REM *** Koenig im Schach ? ***
7410 :
7420 fu = 1
7430 fl = 0
7440 FOR aa=11 TO 88
7450 FOR ba=8 TO 9
7460 IF f(aa) = st * ba THEN pm = aa
7470 NEXT ba, aa
7480 FOR m=3 TO 9
7490 IF fl = 0 THEN GOSUB 7600
7500 NEXT m
```

```
7510 FOR aa=1 TO 2
7520 c = st * ( jp(aa) - (aa-1) * 22 )
7530 IF f(pm + c) = -1 * st THEN fl=1
7540 IF f(pm + c) = -2 * st THEN fl=1
7550 NEXT aa
7560 RETURN
7570 :
7600 REM *** Feldbestimmung ***
7610 :
7620 FOR bb=-1 TO 1 STEP 2
7630 FOR ab=jm(m,1) TO jm(m,2)
7640 c = pm + jp(ab) * nj(m) * bb
7650 pf = pm
7660 pf = pf + jp(ab) * bb
7670 IF pf < 11 OR pf > 88 GOTO 7730
7680 IF SGN(f(pf))=SGN(f(pm)) GOTO 7730
7690 IF f(pf) = 10 GOTO 7730
7700 ON fu GOSUB 7800, 7900, 8200, 8500
7710 IF pf = c GOTO 7730
7720 IF f(pf) = 0 GOTO 7660
7730 NEXT ab
7740 NEXT bb
7750 RETURN
7760 :
7800 REM *** Feldueberpruefung ***
7810 :
7820 IF f(pf) = -m * st THEN fl = 1
7830 RETURN
7840 :
7900 REM *** Zugerzeuger Computerzug **
7910 :
7920 IF m > 2 GOTO 8080
7930 IF pm - pf <> 1 GOTO 8020
7940 IF f(pf) <> 0 GOTO 8110
7950 IF m = 2 GOTO 8080
7960 IF f( pf - 1 ) <> 0 GOTO 8080
7970 fm = fm + 1
```

```
7980 fm(fm, 1) = pm
7990 fm(fm, 2) = pf - 1
8000 GOTO 8080
8010 :
8020 IF pm - pf = 11 GOTO 8040
8030 IF pf - pm <> 9 GOTO 8110
8040 IF pm MOD 10 <> 4 GOTO 8070
8050 IF f( pf + 1 ) <> -1 GOTO 8070
8060 IF f(pf) = 0 THEN 8080 ELSE 8110
8070 IF f(pf) >= 0 GOTO 8110
8080 fm = fm + 1
8090 fm(fm, 1) = pm
8100 fm(fm, 2) = pf
8110 RETURN
8120 :
8200 REM *** Zugerzeuger Spielerzug ***
8210 :
8220 IF m > 2 GOTO 8380
8230 IF pf - pm <> 1 GOTO 8320
8240 IF f(pf) <> 0 GOTO 8400
8250 IF m = 2 GOTO 8380
8260 IF f( pf + 1 ) <> 0 GOTO 8380
8270 IF pm MOD 10 <> 2 GOTO 8380
8280 mm = mm + 1
8290 mm(mm) = pf + 1
8300 GOTO 8380
8310 :
8320 IF pf - pm = 11 GOTO 8340
8330 IF pm - pf <> 9 GOTO 8400
8340 IF pm MOD 10 <> 5 GOTO 8370
8350 IF f( pf - 1 ) <> 1 GOTO 8370
8360 IF f(pf) = 0 THEN 8380 ELSE 8400
8370 IF -f(pf) >= 0 GOTO 8400
8380 mm = mm + 1
8390 mm(mm) = pf
8400 RETURN
8410 :
```

```
8500 REM *** Positionswertung ***
8510 :
8520 IF m > 2 GOTO 8670
8530 IF pm - pf <> sg GOTO 8600
8540 IF f(pf) <> 0 GOTO 8680
8550 IF m = 2 GOTO 8670
8560 IF f( pf - sg ) <> 0 GOTO 8670
8570 v = v + sg * vm(0)
8580 GOTO 8670
8590 :
8600 IF pm - pf = 11 * sg GOTO 8630
8610 IF pf - pm <> 9 * sg GOTO 8680
8620 c0 = 4.5 - 0.5 * sg
8630 IF pm MOD 10 <> c0 GOTO 8660
8640 IF f( pf + sg ) <> -sg GOTO 8660
8650 IF f(pf) = 0 THEN 8670 ELSE 8680
8660 IF f(pf) * sg >= 0 GOTO 8680
8670 v = v + sg * vm( ABS( f(pf) ) )
8680 RETURN
8690 :
8990 REM ******************************
9000 REM *           Spielende          *
9010 REM ******************************
9020 :
9030 WINDOW SWAP 1,0
9040 CLS
9050 PRINT
9060 IF ma = 1 THEN PRINT "PATT!"
9070 IF ma = 2 THEN PRINT "MATT!"
9080 PRINT
9090 RETURN
```

4. Verbesserungsvorschläge für das BASIC-Schachprogramm

4.1. Schachfiguren statt Buchstaben

Die Darstellung der einzelnen Figuren in Form von Groß- und Kleinbuchstaben mag für manche Spieler nicht ganz befriedigend sein. Da die Computer der Firma Schneider über mächtige Grafikbefehle verfügen, eröffnet sich die Möglichkeit, nicht genutzte Zeichen mit Hilfe der SYMBOL AFTER ..-Anweisung aus dem ROM-Bereich in den RAM-Bereich zu kopieren, wo dann die Möglichkeit besteht, die kopierten Zeichen umzudefinieren, denn jedes Zeichen setzt sich aus einer 8 x 8-Punkte-Matrix zusammen, die im RAM-Bereich verändert werden kann.

In der hier vorgeschlagenen Programmverbesserung haben wir mit der SYMBOL-Anweisung jeweils 6 Zeichen dergestalt umdefiniert, daß man daraus anschließend Schachfiguren zusammensetzen kann, die im Hinblick auf die grafische Qualität selbst von den anspruchsvollsten Schachprogrammen in Maschinensprache nicht übertroffen werden.

Zu ergänzende Befehle:

```
1371 :
1372 SYMBOL AFTER 220
1373 FOR a=220 TO 255
1374 READ c0,c1,c2,c3,c4,c5,c6,c7
1375 SYMBOL a, c0,c1,c2,c3,c4,c5,c6,c7
1376 NEXT a
```

```
1377 DATA 0,0,0,0,0,0,1,3,0,0,0,0,0,0
1378 DATA 128,192,7,3,1,31,31,2,2,2,224
1379 DATA 192,128,248,248,64,64,64,2,31
1380 DATA 48,127,0,0,0,0,64,248,12,254
1381 DATA 0,0,0,0
1382 :
1383 DATA 0,0,0,0,3,12,63,63,0,0,0,0
1384 DATA 192,48,184,220,255,127,55,31
1385 DATA 14,14,10,4,238,246,251,251
1386 DATA 122,116,244,244,1,1,3,15,0,0
1387 DATA 0,0,244,250,253,253,0,0,0,0
1388 :
1389 DATA 0,0,0,0,1,3,7,6,0,0,0,0,128
1390 DATA 192,224,96,14,30,048,48,30,14
1391 DATA 6,7,112,120,12,12,120,112,96
1392 DATA 224,3,63,127,199,0,0,0,0,192
1393 DATA 252,254,227,0,0,0,0
1394 :
1395 DATA 0,0,0,0,243,243,243,255,0,0,0
1396 DATA 0,207,207,207,255,127,63,63
1397 DATA 63,63,63,63,48,254,252,252
1398 DATA 252,252,252,252,12,127,64,127
1399 DATA 255,0,0,0,0,254,2,254,255,0,0
1400 DATA 0,0
1401 :
1402 DATA 0,0,0,0,4,78,228,78,0,0,0,0
1403 DATA 32,114,39,114,127,127,96,63
1404 DATA 63,31,31,15,254,254,6,252,252
1405 DATA 248,248,240,15,127,0,255,0,0
1406 DATA 0,0,240,254,0,255,0,0,0,0
1407 :
1408 DATA 0,0,0,0,1,51,123,121,0,0,0,0
1409 DATA 128,204,222,158,140,223,127
1410 DATA 63,63,31,31,15,49,251,254,252
1411 DATA 252,248,248,240,15,127,0,255
1412 DATA 0,0,0,0,240,254,0,255,0,0,0,0
1413 :
```

```
1414 c = 0
1415 gd = CHR$(10) + CHR$(8) + CHR$(8)
1416 FOR a=220 TO 250 STEP 6
1417 c = c + 1
1418 g(c) = " "
1419 FOR b=0 TO 5 STEP 2
1420 g(c) = g(c) + CHR$( a + b )
1421 g(c) = g(c) + CHR$( a + b + 1 )
1422 IF b < 3 THEN g(c) = g(c) + gd
1423 NEXT b
1424 g(c) = g(c) + CHR$(11)
1425 IF c=1 OR c=5 OR c=8 GOTO 1417
1426 NEXT a
1427 :

1450 INK 2, 17
1460 INK 3, 11

6600 PEN SGN( f(pf) ) + 2
6610 :
6620 PRINT g( ABS( f(pf) ) );
```

4.2. Zugeingabe mit dem Joystick

Die Eingabe mit dem Joystick ist bei einem Computer der Firma Schneider besonders einfach zu programmieren, weil der Spezialbefehl JOY(0) zur Verfügung steht, mit dessen Hilfe man den die Eingaben über den Joystick-Port überprüfen kann. In der im folgenden aufgeführten Programmergänzung verwenden wir JOY(0) lediglich, um in die Subroutine "Eingabe durch Joystick" zu verzweigen. Hier "übernehmen" wir dann die Joystick-Eingaben mit der vordefinierten Funktion INKEY$, weil nur so die Möglichkeit besteht, eine evtl. Annullierung der Eingabe aufgrund einer Betätigung der **DEL**-Taste (= CHR$(127)) festzustellen.

Um den Eingabewunsch mit Hilfe des Joysticks zu signalisieren, ist der **Feuer**-Knopf kurz zu betätigen. Anschließend kann man das "blinkende Feld" - ausgehend von der Position A1 - durch Betätigung des Spielhebels auf dem Schachbrett zum ausgesuchten Feld "bewegen". Mit dem **Feuer**-Knopf erklärt man das jeweilige Feld zur Startposition des Zuges. Auf analoge Weise ist die Zielposition einzugeben. Durch nochmaliges Drücken des **Feuer**-Knopfes bestätigt man die Eingabe des Zuges.

Eine unzulässige Zugeingabe hat zur Folge, daß man den Vorgang wiederholen muß.

Zu ergänzende Befehle:

```
2132 IF JOY(0) = 0 GOTO 2140
2134 GOSUB 10000
2136 IF in = CHR$(127) GOTO 2260
2138 GOTO 2220
```

```
2192 IF JOY(0) = 0 GOTO 2200
2194 GOSUB 10000
2196 IF in = CHR$(127) GOTO 2260
2198 GOTO 2220

2305 IF in = "X" GOTO 2320

10000 REM *** Eingabe durch Joystick **
10010 :
10020 in = INKEY$ : REM Eingabepuffer
10030 pf = 11
10040 IF a = 2 THEN pf = pm(1)
10050 LOCATE a * 3 - 2, 4
10060 PRINT CHR$( INT( pf / 10 ) + 64);
10070 PRINT CHR$( pf MOD 10 + 48 );
10080 GOSUB 6500
10090 x = INT( pf / 10 )
10100 y = pf MOD 10
10110 PAPER#3, ( ( x+y+1 ) MOD 2 ) * 2
10120 CLS#3
10130 GOSUB 6500
10140 in = INKEY$
10150 IF in = CHR$(127) GOTO 10270
10160 IF in = "" GOTO 10090
10170 IF INT(pf / 10) < 2 GOTO 10190
10180 IF in = CHR$(8) THEN pf = pf - 10
10190 IF INT(pf / 10) > 7 GOTO 10210
10200 IF in = CHR$(9) THEN pf = pf + 10
10210 IF pf MOD 10 < 2 GOTO 10230
10220 IF in = CHR$(10) THEN pf = pf - 1
10230 IF pf MOD 10 > 7 GOTO 10250
10240 IF in = CHR$(11) THEN pf = pf + 1
10250 IF in <> "X" GOTO 10050
10260 pm(a) = pf
10270 RETURN
10280 :
```

4.3. Blinken der gezogenen Figur

Die Durchführung der Stellungsänderung einer Figur geschieht auf dem Bildschirm so schnell, daß man den Positionswechsel bisweilen nur aufgrund des in WINDOW#2 ausgegebenen Halbzugs erkennen kann.

Die folgenden Befehle lassen die bewegte Figur sowohl auf dem alten als auch auf dem neuen Feld mehrmals aufblinken, so daß man die eingetretene Änderung nicht übersehen kann.

Zu ergänzende Befehle:

```
6040 pf = pm(1)
6043 f1 = f(pf)
6046 f(pf) = f( pm(2) )
6050 GOSUB 20000
6053 f(pf) = f1
6056 GOSUB 6500
6060 pf = pm(2)
6063 GOSUB 20000
6066 GOSUB 6500
6070 IF pm(3) = 0 GOTO 6110
6073 pf = pm(3)
6076 IF pm(4) = 0 THEN f(pf) = -st
6080 IF pm(4) > 0 THEN f(pf) = 5 * st
6083 GOSUB 20000
6086 f(pf) = 0
6090 GOSUB 6500
6093 IF pm(4) = 0 GOTO 6110
6096 pf = pm(4)
6100 GOSUB 20000
6103 GOSUB 6500
```

```
20000 REM *** Blinken von f(pf) ***
20010 :
20020 FOR b=1 TO 5
20030 GOSUB 6500
20040 FOR c=1 TO 90:NEXT
20050 x = INT( pf / 10 )
20060 y = pf MOD 10
20070 PAPER#3, ( ( x+y+1 ) MOD 2 ) * 2
20080 CLS#3
20090 FOR c=1 TO 90:NEXT
20100 NEXT b
20110 RETURN
20120 :
```

4.4. Vorgabe einer Stellung

Oft möchte der Anwender wissen, wie der Rechner auf eine bestimmte Stellung reagiert oder auch nur ein abgebrochenes Spiel fortsetzen. Mit Hilfe der folgenden Programmergänzung kann man Stellungen vorgeben.

Wenn der Anwender aufgrund der Frage des Systems

Stellungsvorgabe (J/N)? _

die **J**-Taste betätigt, erhält man auf der Position A1 ein "blinkendes Feld", das man mit Hilfe der Cursor-Steuertasten (Cursor nach oben: ↑ , Cursor nach rechts: → , Cursor nach unten: ↓ , Cursor nach links: ←) "bewegen" kann.

Das Setzen einer Figur erfolgt durch Betätigung der entsprechenden **Buchstaben-** o d e r **SHIFT- + Buchstabentaste**, beispielsweise "**K**" für "**K**önig des Spielers", "**k**" für "**K**önig des Computers"; "**B**" für "**B**auer des Spielers", "**b**" für "**B**auer des Computers"; ... ; vgl. hierzu Seite 2-05 und 2-06. Eine versehentlich gesetzte Figur kann man mit Hilfe der Leertaste wieder löschen. Die Stellungsvorgabe ist mit der **ENTER**-Taste abzuschließen. Dies ist jedoch nur möglich, wenn beide Könige gesetzt worden sind.

Zu ergänzende Befehle:

```
1705 GOSUB 30000

1965 IF ip = "j" THEN GOSUB 30200
```

```
30000 REM *** Stellungsvorgabe ***
30010 :
30020 CLS
30030 LOCATE 8 ,13
30040 PRINT "Stellungsvorgabe (J/N)? ";
30050 PRINT CHR$(143)
30060 ip = INKEY$
30070 ip = LOWER$(ip)
30080 IF ip = "n" GOTO 30140
30090 IF ip <> "j" GOTO 30060
30100 FOR a=1 TO 8
30110 FOR b=10 TO 80 STEP 10
30120 f( a + b ) = 0
30130 NEXT b, a
30140 RETURN
30150 :
30200 WINDOW SWAP 1,0
30210 pf = 11
30230 LOCATE 2, 4
30240 PRINT CHR$( INT( pf / 10 ) +64 );
30250 PRINT CHR$( pf MOD 10 + 48 );
30260 GOSUB 6500
30270 x = INT( pf / 10 )
30280 y = pf MOD 10
30290 PAPER#3, ( ( x+y+1 ) MOD 2 ) * 2
30300 CLS#3
30310 GOSUB 6500
30320 in = INKEY$
30330 IF in = "" GOTO 30270
30340 IF in = CHR$(13) GOTO 30570
30350 IF INT(pf / 10) < 2 GOTO 30370
30360 IF in = CHR$(242) THEN pf = pf-10
30370 IF INT(pf / 10) > 7 GOTO 30390
30380 IF in = CHR$(243) THEN pf = pf+10
30390 IF pf MOD 10 < 2 GOTO 30410
30400 IF in = CHR$(241) THEN pf = pf- 1
```

```
30410 IF pf MOD 10 > 7 GOTO 30430
30420 IF in = CHR$(240) THEN pf = pf+ 1
30430 IF ASC(in) > 239 GOTO 30220
30440 IF in = "b" THEN f(pf) =  1
30450 IF in = "s" THEN f(pf) =  3
30460 IF in = "l" THEN f(pf) =  4
30470 IF in = "t" THEN f(pf) =  6
30480 IF in = "d" THEN f(pf) =  7
30490 IF in = "k" THEN f(pf) =  9
30500 IF in = "B" THEN f(pf) = -1
30510 IF in = "S" THEN f(pf) = -3
30520 IF in = "L" THEN f(pf) = -4
30530 IF in = "T" THEN f(pf) = -6
30540 IF in = "D" THEN f(pf) = -7
30550 IF in = "K" THEN f(pf) = -9
30560 IF in = " " THEN f(pf) =  0
30570 IF ABS( f(pf) ) <> 1 GOTO 30620
30580 c = 4.5 + f(pf) * 2.5
30590 IF pf MOD 10 = c GOTO 30220
30600 f(pf) = f(pf) * 2
30610 GOTO 30220
30620 IF ABS( f(pf) ) = 9 GOTO 30680
30630 IF ABS( f(pf) ) <> 6 GOTO 30220
30640 c = 14.5 + SGN( f(pf) ) * 3.5
30650 IF pf = c GOTO 30700
30660 c = c + 70
30670 GOTO 30690
30680 c = 54.5 + SGN( f(pf) ) * 3.5
30690 IF pf <> c GOTO 30220
30700 LOCATE 1, 2
30710 PRINT "Schon bewegt?(J/N)"
30720 in = INKEY$
30730 IF UPPER$(in) = "J" GOTO 30760
30740 IF UPPER$(in) <> "N" GOTO 30720
30750 f(pf) = f(pf) - SGN( f(pf) )
30760 CLS
30770 GOTO 30220
```

```
30770 GOTO 30220
30780 fl = 1
30790 FOR a=11 TO 88
30800 IF f(a)= 8 OR f(a)= 9 THEN fl = 0
30810 NEXT a
30820 IF fl = 1 GOTO 30270
30830 fl = 1
30840 FOR a=11 TO 88
30850 IF f(a)=-8 OR f(a)=-9 THEN fl = 0
30860 NEXT a
30870 IF fl = 1 GOTO 30270
30880 WINDOW SWAP 1,0
30885 nm = 16
30890 RETURN
30900 :
```

4.5. Abfrage "Wer soll beginnen?"

Die Ergänzung der in diesem Abschnitt aufgeführten Befehle führt dazu, daß dem Spieler das folgende Menue angeboten wird:

Wer soll beginnen ?

Spieler .... (1)
Computer .... (2)
Zufallswahl .... (3)

.. 1, 2 oder 3 druecken !

Die Betätigung der "**1**"-Taste bewirkt, daß der Spieler anzufangen hat. Das Drücken der "**2**"-Taste hat zur Folge, daß das Schachbrett (nur) auf dem Bildschirm "gedreht" wird, und der Computer beginnt. Da auch in diesem Fall das Funktionsmodul **Computerzug** ohne Einschränkung durchlaufen wird, muß der Spieler - abhängig von der gewählten Spielstärke - geraume Zeit auf die Eröffnung warten. Wir empfehlen daher, die hier aufgeführte Ergänzung nur zusammen mit jener des Kapitels "**4.6. Eröffnungsbibliothek**" zu übernehmen. Wenn man die "**3**"-Taste betätigt, wird die Entscheidung darüber, wer das Spiel zu eröffnen hat, mit Hilfe der Zufallsfunktion **RND** herbeigeführt.

Zu ergänzende Befehle:

```
1695 GOSUB 40000

1725 b = ABS( be / 11 - a )

1740 PRINT CHR$( 57 - b )
```

```
1795 b = ABS( be / 11 - a )

1810 PRINT CHR$( 64 + b ); " ";

1940 :

2245 pm(a) = ABS( be- pm(a) )

5435 ba = ABS( be - pm(aa) )
5440 PRINT CHR$( INT( ba / 10 ) + 64);
5450 PRINT CHR$( ba MOD 10 + 48 );

6295 b = ABS( be - pm(a) )
6300 PRINT CHR$( INT( b / 10 ) + 64);
6310 PRINT CHR$( b MOD 10 + 48 );

6400 IF st = SGN( 1 - be ) THEN PRINT

40000 REM *** Wer beginnt ? ***
40010 :
40020 be = 0
40030 st = -1
40040 CLS
40050 LOCATE 8, 8
40060 PRINT "Wer soll beginnen ?"
40070 LOCATE 8, 12
40080 PRINT "Spieler  ...........  (1)"
40090 LOCATE 8, 14
40100 PRINT "Computer  ..........  (2)"
40110 LOCATE 8, 16
40120 PRINT "Zufallswahl  .......  (3)"
40130 LOCATE 8, 20
40140 PRINT ".. 1, 2 oder 3 druecken !"
40150 LOCATE 9, 24
40160 in = INKEY$
40170 IF in = "1" GOTO 40210
40180 IF in = "2" GOTO 40230
```

```
40190 IF in <> "3" GOTO 40160
40200 IF RND * 2 > 1 GOTO 40230
40210 PRINT "*** Spieler beginnt ***"
40220 GOTO 40300
40230 PRINT "*** Computer beginnt ***"
40240 f(51) = -7
40250 f(41) = -8
40260 f(58) =  7
40270 f(48) =  8
40280 be = 99
40290 st = 1
40300 FOR a=1 TO 2500:NEXT a
40310 RETURN
40320 :
```

Bei Verwendung der Befehle des Kapitels
4.1. Schachfiguren statt Buchstaben
ist außerdem die folgende Ergänzug vorzunehmen:

```
6600 PEN SGN( f(pf) * ( 1 - be ) ) + 2
```

Bei Verwendung der Befehle des Kapitels
4.2. Zugeingabe mit dem Joystick
sind außerdem die folgenden Ergänzungen vorzunehmen:

```
10055 b = ABS( be - pf )
10060 PRINT CHR$( INT( b / 10 ) + 64);
10070 PRINT CHR$( b MOD 10 + 48 );

10260 pm(a) = ABS( be - pf )
```

Bei Verwendung der Befehle des Kapitels
4.4. Vorgabe einer Stellung
sind außerdem die folgenden Ergänzungen vorzunehmen:

```
30235 b = ABS( be - pf )
30240 PRINT CHR$( INT( b / 10 ) + 64 );
30250 PRINT CHR$( b MOD 10 + 48 );
```

4.6. Eröffnungsbibliothek

Es ist empfehlenswert, die ersten Züge eines Schachspiels gesondert zu behandeln, weil sie - im Gegensatz zu den Positionen des Mittelspiels - bekannt sind. Man sollte sie daher nicht berechnen, sondern einer Eröffnungsbibliothek entnehmen, in der in Form von DATA-Anweisungen mehrere Standarderöffnungen gespeichert sind. Eine Eröffnungsbibliothek bietet drei Vorteile:

(1) Sie ermöglicht eine sehr schnelle Reaktion des Rechners.

(2) Sie gewährleistet Erwiderungen des Rechners, die sich aufgrund von Erfahrungen im Schachspiel bewährt haben.

(3) Sie verhindert, daß der Rechner "Eröffnungsfallen" erliegt.

Für unser Programm haben wir 21 bewährte Eröffnungen ausgesucht, die der Leser nach Belieben ergänzen kann. (Zu diesem Zweck muß man nur die entsprechenden DATA-Anweisungen anfügen, den ersten Index der zweiten Variablen und den Index der dritten Variablen des DIM-Befehls in Zeile 1090 sowie die Konstante in Zeile 1430 anpassen.) Jede Eröffnungsvariante besteht hier aus sechzehn Halbzügen.

Zu ergänzende Befehle:

```
1090 DIM mm(27), mb(21,16), no(21)

1430 mb = 21
1431 FOR a=1 TO mb : FOR b=1 TO 16
1432 READ mb(a,b)
1433 no(a) = a
1434 NEXT b, a
1435 :
1436 MODE 1
```

```
2035 nm = nm + 1

4103 GOSUB 50000
4106 IF fl = 1 GOTO 5030

50000 REM *** Eroeffnungsbibliothek ***
50010 :
50020 fl = 0
50030 IF nm > 15 GOTO 50300
50040 co = 0
50050 IF nm = 0 GOTO 50170
50060 pm(1) = ABS( be - pm(1) )
50070 pm(2) = ABS( be - pm(2) )
50080 mv = pm(1) * 100 + pm(2)
50090 FOR a = 1 TO mb
50100 IF mb( no(a),nm) <> mv GOTO 50130
50110 co = co + 1
50120 no(co) = no(a)
50130 NEXT a
50140 IF co = 0 GOTO 50300
50150 mb = co
50160 co = 0
50170 nm = nm + 1
50180 no = no( INT( RND * mb ) + 1 )
50190 FOR a = 1 TO mb
50200 IF mb(no(a),nm)<>mb(no,nm) GOTO 50230
50210 co = co + 1
50220 no(co) = no(a)
50230 NEXT a
50240 mb = co
50250 pm(1) = INT( mb(no,nm) / 100 )
50260 pm(2) = mb(no,nm) MOD 100
50270 pm(1) = ABS( be - pm(1) )
50280 pm(2) = ABS( be - pm(2) )
50290 fl = 1
50300 RETURN
50310 :
```

```
50320 REM Sizilianisch-Rauser
50330 DATA 5254,3735,7163,4746,4244
50340 DATA 3544,6344,7866,2133,2836
50350 DATA 3175,3847,4142,1838,5131
50360 DATA 2644
50370 REM Sizilianisch-Drachen
50380 DATA 5254,3735,7163,4746,4244
50390 DATA 3544,6344,7866,2133,7776
50400 DATA 3153,6877,6152,2836,5171
50410 DATA 5878
50420 REM Spanisch-offen
50430 DATA 5254,5755,7163,2836,6125
50440 DATA 1716,2514,7866,5171,6857
50450 DATA 6151,2725,1423,4746,3233
50460 DATA 5878
50470 REM Giuco piano
50480 DATA 5254,5755,7163,2836,6134
50490 DATA 6835,3233,7866,4244,5544
50500 DATA 3344,3524,3142,2442,2142
50510 DATA 4745
50520 REM Italienische Partie
50530 DATA 5254,5755,7163,2836,6134
50540 DATA 6835,4243,7866,2133,4746
50550 DATA 3175,8786,7566,4866,3345
50560 DATA 6648
50570 REM Zweispringerspiel im Nachzuge
50580 DATA 5254,5755,7163,2836,6134
50590 DATA 7866,6375,4745,5445,3644
50600 DATA 3233,2725,3461,6645,3344
50610 DATA 4875
50620 REM Dreispringerspiel
50630 DATA 5254,5755,7163,2836,2133
50640 DATA 6824,3345,7866,4524,3624
50650 DATA 6355,4857,4244,4746,1213
50660 DATA 4655
50670 REM Philidor-Verteidigung
50680 DATA 5254,5755,7163,4746,4244
50690 DATA 6765,6134,6554,6355,4645
```

```
50700 DATA 4185,7776,5576,7866,8555
50710 DATA 6857
50720 REM Koenigsgambit
50730 DATA 5254,5755,6264,5564,7163
50740 DATA 7775,6134,4746,5171,8786
50750 DATA 4244,6877,3233,2836,7273
50760 DATA 7574
50770 REM Wiener Partie
50780 DATA 5254,5755,2133,7866,6264
50790 DATA 4745,6455,6654,7163,6857
50800 DATA 4244,5878,6143,6765,5566
50810 DATA 5766
50820 REM Franzoesisch-Winawer
50830 DATA 5254,5756,4244,4745,2133
50840 DATA 6824,5455,3735,1213,2433
50850 DATA 2233,7857,4174,3544,7477
50860 DATA 8878
50870 REM Franzoesische Partie
50880 DATA 5254,5756,4244,4745,2133
50890 DATA 7866,3175,6857,5455,6647
50900 DATA 7557,4857,4142,5878,6264
50910 DATA 3735
50920 REM Pirc-Verteidigung
50930 DATA 5254,4746,4244,7866,2133
50940 DATA 7776,3175,6877,6264,3736
50950 DATA 4142,2847,5455,4655,4455
50960 DATA 6678
50970 REM Damengambit-orthodox
50980 DATA 4244,4745,3234,5756,2133
50990 DATA 7866,3175,6835,7163,5878
51000 DATA 5253,2847,1131,3736,6143
51010 DATA 4534
51020 REM Nimzo-Indisch-Rubinstein
51030 DATA 4244,7866,3234,5756,2133
51040 DATA 6824,5253,2726,6143,3827
51050 DATA 7163,2433,2233,4745,3113
51060 DATA 2847
```

```
51070 REM Benoni-Verteidigung
51080 DATA 4244,7866,3234,3735,4445
51090 DATA 5755,2133,4746,5254,6857
51100 DATA 7152,1716,5273,7776,6143
51110 DATA 8785
51120 REM Koenigsindisch-kassisch
51130 DATA 4244,7866,3234,7776,2133
51140 DATA 6877,5254,4746,6152,5878
51150 DATA 7163,5755,5171,2847,4445
51160 DATA 4735
51170 REM Koenigsindisch Awerbach-Syst.
51180 DATA 4244,7866,3234,7776,2133
51190 DATA 6877,5254,4746,6152,5878
51200 DATA 3175,2847,4142,5755,7163
51210 DATA 3736
51220 REM Koenigsindisch Saemisch-Syst.
51230 DATA 4244,7866,3234,7776,2133
51240 DATA 6877,5254,4746,6152,5878
51250 DATA 3153,5755,4445,3736,4142
51260 DATA 3645
51270 REM Englisch-symmetrisch
51280 DATA 3234,3735,2133,2836,7273
51290 DATA 7776,6172,6877,7163,7866
51300 DATA 5171,5878,4244,3544,6344
51310 DATA 3644
51320 REM Sizilianisch im Anzuge
51330 DATA 3234,5755,2133,7866,7273
51340 DATA 4745,3445,6645,6172,4533
51350 DATA 2233,2836,4243,6835,7163
51360 DATA 5878
```

Bei Verwendung der Befehle des Kapitels
4.4. Vorgabe einer Stellung
ist außerdem die folgende Ergänzung vorzunehmen:

```
30885 nm = 16
```

4.7. Veränderung der Stellungsbewertung

Eine Veränderung der Stellungsbewertung geschieht in erster Linie durch Zuordnung anderer Figurenwerte in den Befehlszeilen 1120 und 1130. Verringert man beispielsweise den Wert der Dame von 50 auf 30, so hat dies zur Folge, daß der Rechner eher bereit ist, im Rahmen eines Zuges seine Dame zu opfern.

Außerdem ist es möglich, das Gewicht der leeren Felder, die besetzt werden können, zu verändern. Würde man z. B. die Konstante 1 in der Zeile 1120 auf 6 heraufsetzen, so wäre ein leeres Feld im Rahmen der Wertung höher gewichtet als ein Bauer, der geschlagen werden kann. Dies hätte zur Folge, daß das Spiel weniger aggressiv geführt wird.

Interessante Ausbaumöglichkeiten der Bewertungsfunktion bestehen **u. a.** darin,

- Figuren auf zentralen Feldern höher zu veranschlagen als in Randbereichen,

- Figuren, die sich gegenseitig decken, höher zu bewerten,

- zusätzliche Punkte für die Ausführung einer Rochade zu vergeben,

- den Bauern in der Nähe der gegnerischen Grundlinie wesentlich höher zu bewerten,

- für die Umwandlung eines Bauern in eine Dame auf der Grundlinie des Gegners zusätzliche Punkte zu vergeben, die noch über den Wert der Dame hinausgehen.

4.8. Einführung zusätzlicher Spielstärken

Während die Spielqualität des Computers bei Spielstärke 1 den fortgeschrittenen Anwender im Regelfall wohl nicht befriedigen wird, hat die bisherige Spielstärke 2 den entscheidenden Nachteil, daß die Antwortzeiten häufig unerträglich lang werden. Eine Verbesserung des Programms könnte man deshalb durch das Angebot zusätzlicher Spielstärken erreichen, die qualitativ und im Hinblick auf die Reaktionszeiten zwischen Spielstärke 1 und 2 liegen. Dies ließe sich vor allem dadurch erreichen, daß man die zu untersuchenden fiktiven Gegenzüge des Spielers z. B. auf die 2 / 3 / 5 (vorläufig) besten Halbzüge des Computers (= **neue Spielstärken 2 / 3 / 4**) beschränkt. Die ehemalige Spielstärke 2, bei der wir zu allen zulässigen Halbzügen des Computers alle fiktiven Halbzüge des Spielers ermittelten und bewerteten, wird hier zur **Spielstärke 5** umfunktioniert.

Zu ergänzende Befehle:

```
1142 FOR a=1 TO 5 : READ va(a) : NEXT a
1144 DATA 0, 2, 3, 5, 120
1146 :

1530 LOCATE 7, 9

1550 :
1560 FOR a=1 TO 5
1570 LOCATE 7, ( a * 2 ) + 11
1580 PRINT "Spielstaerke";
1590 PRINT a;
1600 PRINT " ...... (";
1610 PRINT USING "#"; a;
```

```
1620 PRINT ")"
1630 NEXT a
1640 :
1650 LOCATE 8, 24
1660 PRINT "... Zifferntaste druecken!"

1680 IF in < "1" OR in > "5" GOTO 1670

4485 IF fm > va(mo) THEN fm = va(mo)
```

5. Überprüfung des abgeschriebenen Programms

Nachdem man das BASIC-Schachprogramm eingegeben hat, sollte es sorgfältig überprüft werden, da nur bei vollkommener Fehlerfreiheit der Befehlszeilen ein einwandfreier Programmlauf gewährleistet ist. Wir empfehlen daher, eine Abschreibkontrolle mit einer zweiten Person durchzuführen: Die erste liest das Programm vom Bildschirm vor, und die zweite kontrolliert es im Buch (n i c h t u m g e k e h r t !). Darüber hinaus ist nicht selten eine gewisse Fehlerlokalisierung mit den in den folgenden Abschnitten dargestellten Maßnahmen möglich.

5.1. Überprüfung des Funktionsmoduls Initialisierung

Zur Überprüfung des Funktionsmoduls Initialisierung empfehlen wir die folgenden Maßnahmen:

(1) Starten der Grundversion des Programms (= Kapitel 2 bzw. 3) mit dem Kommando **RUN** und

(2) Auswahl der Spielstärke 1 nach Erscheinen des entsprechenden Menue-Bildes.

(3) Danach muß das Schachbrett mit "Figuren" in der Grundstellung auf dem Bildschirm erscheinen, wie es auf Seite 2-06 dargestellt wurde. In jedem Fall sollte man jetzt durch zweimalige Betätigung der **ESC**-Taste die Programmausführung unterbrechen.

(4) Bei der Wiedergabe des Schachbretts können u. a. folgende Fehler auftreten:

- Einzelne Figuren sind falsch gesetzt; dann vor allem die Zeilen 1150 - 1250 überprüfen!

- Alle Figuren einer bestimmten "Figurenart" werden durch ein falsches Zeichen repräsentiert, bzw. auf allen Leerfeldern erscheint ein bestimmtes Zeichen; dann zuerst die Zeilen 1400 - 1410 bzw. 6620 kontrollieren!

- Es erscheint eine falsche Randbeschriftung; dann zunächst die Zeilen 1720 - 1830 vergleichen!

- Die einzelnen Schachfelder werden fehlerhaft ausgegeben; dann die Zeilen 6520 - 6630 untersuchen!

Wenn das Schachbrett mit den "Figuren" korrekt ausgegeben wird, sind weitere Überprüfungen empfehlenswert:

(5) Eingabe des Kommandos **MODE 2,**

(6) Eingabe der folgenden Befehlszeile im direkten Modus (d. h. ohne Zeilennummer):
RESTORE : FOR B=1 TO 145 : READ C : D=D+C : NEXT : PRINT D
Es muß die Zahl **4680** erscheinen. Falls das System einen anderen Wert ausgibt, sollte man zunächst die Zeilen 1120 - 1130, 1160 - 1250, 1280, 1310, 1360 - 1370 überprüfen!

(7) Eingabe der folgenden Befehlszeile im direkten Modus:
RESTORE 1400 : FOR B=0 TO 18 : READ G : PRINT G; "-"; : NEXT
Folgende Zeichen müssen dann auf dem Bildschirm erscheinen:
K-K-D-T-T-L-S-B-B--b-b-s-l-t-t-d-k-k-
Ist dies nicht der Fall, sind die Zeilen 1400 - 1410 zu überprüfen!

5.2. Überprüfung des Funktionsmoduls Spielereingabe

Zur Überprüfung des Funktionsmoduls Spielereingabe ist folgendermaßen zu verfahren:

(1) Ergänzung einer Programmzeile, die im Anschluß an die Überprüfungsmaßnahmen wieder zu entfernen ist:
2475 STOP

(2) Starten der Grundversion des Programms (= Kapitel 2 bzw. 3) mit dem Kommando **RUN**,

(3) Auswahl der Spielstärke 1 nach Erscheinen des entsprechenden Menue-Bildes und

(4) Eingabe des Zuges **E1-G1** (nicht vergessen, die Eingabe durch Betätigung der **ENTER**-Taste abzuschließen!).

(5) Wenn der gewünschte Zug (= unzulässige Rochade) nicht durch Löschung der Zeichen **E1-G1** zurückgewiesen wird, sollte man vor allem die Zeilen 7120 - 7310 überprüfen!

(6) Eingabe des Zuges **E2-E4**; nachdem die Nachricht
Break in 2475
Ready
ausgegeben worden ist, sollte man die folgenden Kommandos im direkten Modus eingeben:

- **PRINT PM(1); PM(2)**
  Es müssen dann die Zahlen
  52 54
  erscheinen, andernfalls sind vor allem die Zeilen 2170 und 2240 zu vergleichen!

- **PRINT RO**
  Es muß dann die Zahl

0
erscheinen, andernfalls sind in erster Linie die Zeilen 2340 - 2370 und 7120 - 7310 zu kontrollieren!

- **PRINT JP**
  Es muß anschließend die Zahl
  1
  erscheinen, andernfalls sind vor allem die Zeilen 3020 - 3110 zu untersuchen!

Wenn man allerdings mit der in Zeile 2475 eingefügten STOP-Anweisung keine Programmunterbrechung bewirkt, dann ist der Fehler in erster Linie in den Zeilen 3020 - 3720 zu suchen!

5.3. Überprüfung des Funktionsmoduls Computerzug

(1) Starten der Grundversion des Programms (= Kapitel 2 bzw. 3) mit dem Kommando **RUN**,

(2) Auswahl der Spielstärke 1 nach Erscheinen des entsprechenden Menue-Bildes und

(3) Eingabe des Zuges **E2-E4** (nicht vergessen, die Eingabe durch Betätigung der **ENTER**-Taste abzuschließen!).

(4) Es ist jetzt der Gegenzug des Computers abzuwarten. (Da die Grundversion des Schachprogramms noch nicht die Eröffnungsbibliothek enthält - vgl. hierzu Kapitel 4.6. ! - muß der Computer seinen Gegenzug erst berechnen.)

Nachdem der Computer seinen Zug ausgeführt und im WINDOW#2 ausgegeben hat, unterbrechen wir den Programmlauf durch zweimalige Betätigung der **ESC**-Taste und geben anschließend das Kommando **MODE 2** ein.

(5) Wir geben jetzt im direkten Modus einen Schleifenbefehl ein:
FOR A=1 TO FM : PRINT FM(A,1), FM(A,2), FM(A,3) : NEXT
Daraufhin sollten die folgenden Zahlen erscheinen:

57	55	0
57	56	0
47	45	-2
47	46	-3
78	66	-5
37	35	-8
28	36	-9
17	15	-9
37	36	-9
77	75	-9
77	76	-9
27	26	-9
67	65	-10
67	66	-11
78	86	-11
87	86	-11
27	25	-12
87	85	-13
17	16	-15
28	16	-25

Wenn Abweichungen festgestellt werden, sind vor allem die Zeilen 4110 - 4260, 7620 - 7740 und 7920 - 8100 einer Kontrolle zu unterziehen!

Wenn die Reihenfolge der Zahlen in der 3. Spalte nicht zutreffend ist, vermuten wir den Fehler in den Zeilen 5640 - 5720; wenn hingegen andere Zahlen als die vorgegebenen erscheinen, sollte man den Fehler zuerst in den Zeilen 5820 - 5950 und 8520 - 8670 suchen!

Wenn mehr als 20 x 3 Zahlen erscheinen, sind vor allem die Zeilen 7620 - 7740 zu vergleichen!

Wenn Abweichungen bei den Zahlen in der 1. Spalte festgestellt werden, sind in erster Linie die Zeilen 4110 - 4260 zu kontrollieren!

LITERATURHINWEISE

Die folgenden Bücher können in jeder Buchhandlung bestellt und ggf. kurzfristig beschafft werden.

Bartel, Rainer / Kraas, Hans-Joachim / Schrüfer, Günther:
"Das große Computerschach-Buch",
Verlag Data Becker, Düsseldorf 1985,
ISBN: 3-89011-117-3
(Programmiersprache: BASIC und 6510-Maschinensprache)

Hamann, Günter O. / Eden, Jan-Jürgen:
"Mit BASIC ran ans Schachprogramm für Commodore 64
(und VC 20 mit 28 K RAM)", Taschenbuch,
Deutscher Betriebswirte-Verlag GmbH, Gernsbach 1984,
ISBN: 3-88640-019-0
(Programmiersprache: BASIC)

Hamann, Günter O. / Eden, Jan-Jürgen:
"Schachprogrammierung in BASIC mit SHARP MZ-800, SHARP MZ-700",
Deutscher Betriebswirte-Verlag GmbH, Gernsbach 1985,
ISBN: 3-88640-029-8
(Programmiersprache: BASIC)

Ketterling, Hans-Peter / Schwenkel, Frieder / Weiner, Ossi:
"Schach dem Computer",
Goldmann-Taschenbuch 10861, München 1983,
ISBN: 3-442-10861-6
(Das Buch enthält nur allgemeine Hinweise zur Logik der Schachprogrammierung und zu den üblichen Schachcomputern. Es ist z. Zt. vergriffen und steht dem Leser deshalb nur über öffentliche Bibliotheken, ggf. per Fernleihe, zur Verfügung.)

Spracklen, Dan and Kathe:
"Sargon: A Computer Chess Program",
Hayden Book Company, Inc., Rochelle Park, New Jersey, 1978,
ISBN: 0-8104-5155-7
(Programmiersprache: Z80-Assembler)

White, John:
"Schach und andere Strategiespiele auf dem Commodore 64",
Commodore Sachbuchreihe, Band 17,
Commodore Büromaschinen GmbH, Frankfurt/M. 1985,
ISBN: 3-89133-017-0
(Programmiersprache: BASIC und Maschinensprache)

Die folgenden Aufsätze können im Bedarfsfalle durch öffentliche Bibliothenken - ggf. per Fernleihe in Form von Fotokopien - beschafft werden.

Frey, P. W. / Atkin, L. R.:
"Creating a chess player, Part 1, 2, 3, 4",
in: BYTE 3 (1978), Heft 10, Seite 182 - 191, Heft 11, Seite 162 - 181, Heft 12, Seite 140 - 157,
BYTE 4 (1979), Heft 1, Seite 126 - 145
(Programmiersprache: Pascal)

Hamann, Günter O. / Eden, Jan-Jürgen:
"Schachfiguren statt Buchstaben
- BASIC-Programmroutine zur Erzeugung eines Schachbretts mit Schachfiguren für den Commodore 64 -",
in: Europa-Rochade, Okt. 1985, Nr. 10, Seite 16 - 18
(Programmiersprache: BASIC / Der Aufsatz bietet eine Ergänzung zu dem aufgeführten Buch "Mit BASIC ran ans Schachprogramm für Commodore 64" von Hamann, Günter O. / Eden, Jan-Jürgen.)

Steinwender, Dieter:
"Wir schreiben ein Schachprogramm, Teil 1, 2, 3, 4, 5",
in: Computer-Schach & Spiele, 1984, Heft 3, Seite 26 - 32, Heft 4 - 5, Seite 40 - 47, Heft 6, Seite 20 - 25,
Computer-Schach & Spiele, 1985, Heft 1, Seite 14 - 20, Heft 2, Seite 20 - 28
(Programmiersprache: BASIC)

STICHWORTVERZEICHNIS

DBV

G.O. Hamann · J.-J. Eden

BASIC-Schachprogramm schreiben mit Schneider

AMSTRAD CPC

OCR

MÉMOIRE ÉCRITE
MEMORY ENGRAVED
MEMORIA ESCRITA

https://acpc.me/